MITSUBISHI

三菱可编程控制器

MELSEC Q 系列　　MELSEC L 系列

MELSEC-Q/L结构体编程手册

公共指令篇

● 安全注意事项 ●

（使用之前请务必阅读）

在使用 MELSEC-Q 系列、MELSEC-L 系列可编程控制器之前，应仔细阅读各产品附带的手册及附带手册中所介绍的关联手册，同时在充分注意安全的前提下正确地操作。

请妥善保管产品附带手册，放置于操作人员易于取阅的地方，并应将本手册交给最终用户。

● 关于产品的应用 ●

(1) 在使用三菱可编程控制器时，应该符合以下条件 ： 即使在可编程控制器设备出现问题或故障时也不会导致重大事故，并且应在设备外部系统地配备能应付任何问题或故障的备用设备及失效保险功能。

(2) 三菱可编程控制器是以一般工业用途等为对象设计和制造的通用产品。因此，三菱可编程控制器不应用于以下设备・系统等特殊用途。如果用于以下特殊用途，对于三菱可编程控制器的质量、性能、安全等所有相关责任（包括但不限于债务未履行责任、瑕疵担保责任、质量保证责任、违法行为责任、制造物责任），三菱将不负责。

- 面向各电力公司的核电站以及其它发电厂等对公众有较大影响的用途。
- 用于各铁路公司或公用设施目的等有特殊质量保证体系要求的用途。
- 航空航天、医疗、铁路、焚烧・燃料装置、载人移动设备、载人运输装置、娱乐设备、安全设备等预计对人身财产有较大影响的用途。

然而，对于上述应用，如果在限定于具体用途，无需特殊质量（超出一般规格的质量等）要求的条件下，经过三菱的判断也可以使用三菱可编程控制器，详细情况请与当地三菱代表机构协商。

修订记录

＊本手册号在封底的左下角。

印刷日期	＊手册编号	修改内容
2010 年 04 月	SH(NA)-080904CHN-A	第一版

日文手册原稿： SH-080736-F

前言

在此感谢贵方购买了三菱 MELSEC-Q、MELSEC-L 系列通用可编程控制器。

在使用之前应熟读本书，在充分了解 MELSEC 系列可编程控制器的功能・性能的基础上正确地使用本产品。

此外，应将本手册交给最终用户。

目录

关于手册

在 GX Works2 中，根据要使用的功能，将相关手册以分册形式发行。

关联手册

与本产品有关的手册如下所示。

请根据需要参考本表订购。

(1) 结构体编程

手册名称	手册编号
MELSEC-Q/L/F 结构体编程手册（基础篇） 对结构体程序创建中必要的编程方法、编程语言的种类等有关内容进行说明。 （另售）	SH-080903CHN
MELSEC-Q/L 结构体编程手册（应用函数篇） 对结构体程序中可使用的应用函数相关的规格、功能等有关内容进行说明。 （另售）	SH-080905CHN
MELSEC-Q/L 结构体编程手册（特殊指令篇） 对结构体程序中可使用的模块专用指令、PID 控制指令以及内置 I/O 功能用用指令等的特殊指令相关的规格、功能等有关内容进行说明。 （另售）	SH-080906CHN

(2) GX Works2 的操作

手册名称	手册编号
GX Works2 Version1 操作手册（公共篇） 对 GX Works2 的系统配置及参数设置、在线功能的操作方法等、简易工程及结构体工程的通用功能等有关内容进行说明。 （另售）	SH-080932CHN
GX Works2 Version1 操作手册（结构体工程篇） 对 GX Works2 的结构体工程中的程序创建、监视等的操作方法等有关内容进行说明。 （另售）	SH-080934CHN
GX Works2 入门指南（结构体工程篇） 对初次使用 GX Works2 的用户介绍结构体工程中的程序创建及编辑、监视、调试的基本操作方法等有关内容进行说明。 （另售）	SH-080936CHN

要 点

各操作手册以 PDF 文件被存储在软件包的 CD-ROM 中。 备有用于另售的印刷品，希望单独购买手册的情况下，请通过上述表格中的手册编号购买。

1 概要

1.1 本手册的定位

在本手册中，对结构体程序创建中使用的公共指令的有关内容进行说明。

以目的进行分类的参阅手册如下所示。

关于各手册的记载内容、手册编号等请参阅 “关联手册”的列表。

(1) GX Works2 的操作

目的		GX Works2 安装步骤说明书	GX Works2 入门		GX Works2 Version1 操作手册			
								
		-	简易工程篇	结构体工程篇	公共篇	简易工程篇	结构体工程篇	智能功能模块操作篇
安装	希望了解运行环境、安装方法	详细						
简易工程的操作	希望了解基本操作及步骤		详细		概要	概要		
	希望了解编程用的功能及操作方法				概要	详细	详细 *1	
	希望了解除编程以外的所有功能及操作方法				详细			
结构体工程的操作	希望了解基本操作及步骤			详细	概要		概要	
	希望了解编程用的功能及操作方法				概要	详细	详细	
	希望了解除编程以外的所有功能及操作方法				详细			
智能功能模块的操作	希望了解智能功能模块的数据设置方法							详细

*1: 仅 ST 程序。

(2) 编程

目的		MELSEC-Q/L/F 结构体编程手册	MELSEC-Q/L 结构体编程手册			MELSEC-Q/L 编程手册	MELSEC-Q/L/QnA 编程手册	智能功能模块用户手册 / 网络模块参考手册
		基础篇	公共指令篇	特殊指令篇	应用函数篇	公共指令篇	PID 控制指令篇	-
简易工程中的编程	希望了解公共指令的种类及详细内容、出错代码、特殊继电器・特殊寄存器的内容					详细		
	希望了解智能功能模块用指令的种类及详细内容							详细
	希望了解网络模块用指令的种类及详细内容							详细
	希望了解 PID 控制用指令的种类及详细内容						详细	
结构体工程的编程	希望了解初次进行结构体编程的基础知识	详细						
	希望了解公共指令的种类及详细内容		详细					
	希望了解智能功能模块用指令的种类及详细内容			详细				详细
	希望了解网络模块用指令的种类及详细内容			详细				详细
	希望了解 PID 控制用指令的种类及详细内容			详细			详细	
	希望了解出错代码、特殊继电器・特殊寄存器的内容					详细		
	希望了解应用函数的种类及详细内容				详细			

1.2 本手册中使用的总称・略称

在本手册中，将软件包、可编程控制器 CPU 等以如下所示的总称・略称表示。在需要标明相关型号的情况下，将记载模块型号。

总称 / 略称	总称・略称的内容
GX Works2	产品型号 SWnDNC-GXW2 的总称产品名。(n= 版本)
基本型 QCPU	Q00J,Q00,Q01 的总称。
高性能型 QCPU	Q02, Q02H, Q06H, Q12H, Q25H 的总称。
通用型 QCPU	Q00UJ, Q00U, Q01U, Q02U, Q03UD, Q03UDE, Q04UDH, Q04UDEH, Q06UDH, Q06UDEH, Q10UDH, Q10UDEH, Q13UDH, Q13UDEH, Q20UDH, Q20UDEH, Q26UDH, Q26UDEH 的总称。
QCPU(Q 模式)	基本型 QCPU、高性能型 QCPU、通用型 QCPU 的总称。
LCPU	L02CPU、L26-BTCPU 的总称。
CPU 模块	QCPU(Q 模式)、LCPU 的总称。
QnCPU	Q00J, Q00, Q01, Q02 的总称。
QnHCPU	Q02H, Q06H, Q12H, Q25H 的总称。
QnUCPU	Q00UJ, Q00U, Q01U, Q02U, Q03UD, Q03UDE, Q04UDH, Q04UDEH, Q06UDH, Q06UDEH, Q10UDH, Q10UDEH, Q13UDH, Q13UDEH, Q20UDH, Q20UDEH, Q26UDH, Q26UDEH 的总称。
QnU(D)(H)CPU	Q02U, Q03UD, Q04UDH, Q06UDH, Q10UDH, Q13UDH, Q20UDH, Q26UDH 的总称。
QnUD(H)CPU	Q03UD, Q04UDH, Q06UDH, Q10UDH, Q13UDH, Q20UDH, Q26UDH 的总称。
QnUDE(H)CPU	Q03UDE, Q04UDEH, Q06UDEH, Q10UDEH, Q13UDEH, Q20UDEH, Q26UDEH 的总称。
CC-Link IE 控制网络	CC-Link IE 控制网络系统的略称。
MELSECNET/H	MELSECNET/H 网络系统的略称。
CC-Link	Control & Communication Link 的略称。
个人计算机	基于 Windows® 的个人计算机的总称。
公共指令	顺控程序指令、基本指令、应用指令、数据链接指令、多 CPU 专用指令、多 CPU 高速通信专用指令的总称。
特殊指令	模块专用指令、PID 控制指令、Socket(套接字) 通信功能用指令、内置 I/O 功能用指令、数据记录功能用指令的总称。

2 指令列表

2.1 指令的分类

CPU 模块的指令大致分为顺控程序指令、基本指令、应用指令、多 CPU 间专用指令、多 CPU 间高速通信专用指令。此外，指令的分类如表 2.1 所示。

表 2.1 指令的分类

指令的分类		内容	说明章节
顺控程序指令	触点指令	运算开始、串联连接、并联连接。	5 章
	合并指令	梯形图块的连接、运算结果的脉冲化、运算结果的存储・读取。	
	输出指令	位软元件的输出、脉冲输出、输出取反。	
	移动指令	位软元件的移动。	
	主控指令	主控制	
	结束指令	程序的结束。	
	其它指令	程序的停止、无处理等上述分类中未列入的指令。	
基本指令	比较运算指令	=, ＞, ＜等的比较。	6 章
	算术运算指令	BIN、BCD 的加减乘除。	
	BCD↔BIN 转换指令	BCD→BIN、BIN→BCD 的转换。	
	数据传送指令	指定的数据的传送。	
	程序分支指令	程序的跳转。	
	程序的执行控制指令	中断程序的允许 / 禁止。	
	I/O 刷新指令	部分刷新的执行。	
	其它方便指令	增减计数器、示教定时器、特殊功能定时器、旋转台的就近控制等的指令。	
应用指令	逻辑运算指令	逻辑和、逻辑积等的逻辑运算。	7 章
	旋转指令	指定的数据的旋转。	
	移动指令	指定的数据的移动。	
	位处理指令	位设置 / 复位、位测试、位软元件的批量复位。	
	数据处理指令	16 位数据的查找、解码、编码等的数据处理。	
	结构体指令	循环运算、子程序的调用、梯形图单位的变址修饰。	
	表操作指令	数据表的读取 / 写入。	
	缓冲存储器访问指令	智能功能模块的数据读取 / 写入。	
	显示指令	ASCII 码的打印、字符的 LED 显示等。	
	调试・故障诊断指令	检查、状态锁存，采样跟踪、程序跟踪。	
	字符串处理指令	BIN/BCD↔ASCII 转换、BIN↔字符串转换、浮动小数点数据↔字符串转换、字符串处理等。	
	特殊函数指令	三角函数、度弧度转换、指数运算、自然对数、常用对数、平方根。	
	数据控制指令	上下限极限控制、死区控制、区域控制、标度。	
	切换指令	文件寄存器的块号切换、文件寄存器 / 注释文件指定。	
	时钟用指令	年、月、日、时、分、秒、星期的读取 / 写入，时、分、秒的加减法运算，时、分、秒↔秒的转换，年、月、日的比较，时、分、秒的比较。	
	扩展时钟用指令	年、月、日、时、分、秒、1/1000 秒、星期的读取，时、分、秒、1/1000 秒的加减法运算。	
	程序控制用指令	程序的执行条件的切换指令。	
	其它指令	WDT 复位、定时时钟等上述分类中未列入的指令。	
数据链接用指令	链接刷新用指令	指定网络的刷新。	8 章
	路由信息读取 / 写入指令	路由信息的读取 / 写入。	
多 CPU 间专用指令	多 CPU 间专用指令	至本站 CPU 共享存储器的写入、来自于其它机号 CPU 共享存储器的读取	9 章
多 CPU 间高速通信专用指令	多 CPU 间软元件写入 / 读取指令	至其它机号的软元件写入、来自于其它机号中的软元件读取	10 章

2.2 指令列表的阅读方法

2.3 ～ 2.5 节的指令列表的格式如下所示。

表 2.2 指令列表的阅读方法

分类	指令名	自变量	处理内容	执行条件	说明页面
跳转	CJ	ⓟ	•输入条件成立时跳转至ⓟ。		6-137
	SCJ	ⓟ	•从输入条件成立的下一个扫描开始跳转至ⓟ。		
	JMP	ⓟ	•无条件跳转至ⓟ。		
1)	2)	3)	4)	5)	6)

说明

1) ..... 将指令按用途进行分类。

2) ..... 表示程序中使用的指令。

3) ..... 表示指令的自变量。

ⓢ, ⓢ1 ..... 源 : 运算前存储数据。

ⓓ, ⓓ1 ..... 目标 : 表示运算后的数据的去向。

n, n1 ..... : 对软元件数 / 传送数进行指定。

4) ..... 表示各指令的处理内容。

5) ..... 各指令的执行条件，其详细内容如下所示。

符号	执行条件
未记入	是常时执行的指令，与指令前条件的 ON/OFF 无关，常时执行。 前条件为 OFF 的情况下，该指令执行 OFF 处理。
	是 ON 中执行型的指令，只有在指令的前条件为 ON 期间执行该指令。前条件为 OFF 的的情况下，不执行该指令，不进行处理。
	是 ON 时 1 次执行型的指令，只有在指令的前条件的上升沿时 (OFF → ON) 执行指令，以后即使条件为 ON 也不执行该指令，不进行处理。
	是 OFF 中执行型的指令，只有在指令的前条件为 OFF 期间执行该指令。前条件为 ON 的情况下，不执行该指令，不进行处理。
	是 OFF 时 1 次执行型的指令，只有在指令的前条件的下降沿时 (ON → OFF) 执行指令，以后即使条件为 OFF 也不执行该指令，不进行处理。

6) ..... 表示说明各指令的页面。

2.3 顺控程序指令

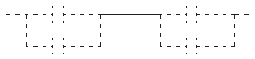
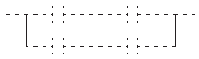
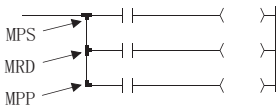
2.3.1 触点指令

表 2.3 触点指令

分类	指令名	结构体梯形图中的符号	处理内容	执行条件	说明页面
触点	LD		• 逻辑运算开始 （常开触点逻辑运算开始）		5-2
	LDI		• 逻辑否定运算开始 （常闭触点逻辑运算开始）		
	AND		• 逻辑积（常开触点串联连接）		
	ANDI		• 逻辑积否定（常闭触点串联连接）		
	OR		• 逻辑和（常开触点并联连接）		
	ORI		• 逻辑和否定（常闭触点并联连接）		
	LDP		• 上升沿脉冲运算开始		5-5
	LDF		• 下降沿脉冲运算开始		
	ANDP		• 上升沿脉冲串联连接		
	ANDF		• 下降沿脉冲串联连接		
	ORP		• 上升沿脉冲并联连接		
	ORF		• 下降沿脉冲并联连接		
	LDPI		• 上升沿脉冲否运算开始		5-8
	LDFI		• 下降沿脉冲否运算开始		
	ANDPI		• 上升沿脉冲否串联连接		
	ANDFI		• 下降沿脉冲否串联连接		
	ORPI		• 上升沿脉冲否并联连接		
	ORFI		• 下降沿脉冲否并联连接		

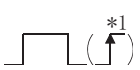
2.3.2 合并指令

表 2.4 合并指令

分类	指令名	结构体梯形图中的符号	处理内容	执行条件	说明页面
合并	ANB		• 逻辑块间的 AND （逻辑块间的串联连接）		5-11
	ORB		• 逻辑块间的 OR （逻辑块间的并联连接）		
	MPS		• 运算结果的存储		5-13
	MRD		• MPS 中存储的运算结果的读取		
	MPP		• MPS 中存储的运算结果的读取及复位		
	INV	-	• 运算结果的取反		5-16
	MEP	-	• 运算结果上升沿脉冲化		5-18
	MEF	-	• 运算结果下降沿脉冲化		
	EGP	-	• 运算结果上升沿脉冲化		5-20
	EGF	-	• 运算结果下降沿脉冲化		

2.3.3 输出指令

表 2.5 输出指令

分类	指令名	自变量	处理内容	执行条件	说明页面
输出	OUT	ⓓ	• 软元件的输出		5-22 5-24 5-28 5-31
	SET	ⓓ	• 软元件的设置		5-33 5-38
	RST	ⓓ	• 软元件的复位		5-35 5-38
	PLS	ⓓ	• 输入信号的上升沿时程序发生 1 个周期的脉冲。		5-41
	PLF	ⓓ	• 输入信号的下降沿时程序发生 1 个周期的脉冲。		
	FF	ⓢ	• 软元件输出的取反		5-44
	DELTA	ⓓ	• 直接输出的脉冲化		5-46
	DELTAP	ⓓ			

*1 ： 只有使用报警器 (F) 时变为  。

2.3.4 移动指令

表 2.6 移动指令

分类	指令名	自变量	处理内容	执行条件	说明页面
移动	SFT	ⓓ	• 软元件的 1 位移动		5-48

2.3.5 主控指令

表 2.7 主控指令

分类	指令名	自变量	处理内容	执行条件	说明页面
主控	MC	n*, ⓓ	• 主控开始		5-51
	MCR	n*	• 主控解除		

2.3.6 结束指令

表 2.8 结束指令

分类	指令名	自变量	处理内容	执行条件	说明页面
程序结束	FEND	-	• 主程序的结束		5-55

2.3.7 其它指令

表 2.9 其它指令

分类	指令名	自变量	处理内容	执行条件	说明页面
停止	STOP	-	• 输入条件成立后，停止顺控程序的运算。 • 将 RUN/STOP（键）开关再次置为 RUN 时，执行顺控程序。		5-57

2.4 基本指令

2.4.1 比较运算指令

表 2.10 比较运算指令

分类	指令名	自变量	处理内容	执行条件	说明页面
16 位数据比较	LD=	ⓢ1, ⓢ2	•ⓢ1 = ⓢ2 时导通状态 •ⓢ1 ≠ ⓢ2 时非导通状态		6-2
	AND=	ⓢ1, ⓢ2			
	OR=	ⓢ1, ⓢ2			
	LD<>	ⓢ1, ⓢ2	•ⓢ1 ≠ ⓢ2 时导通状态 •ⓢ1=ⓢ2 时非导通状态		
	AND<>	ⓢ1, ⓢ2			
	OR<>	ⓢ1, ⓢ2			
	LD<=	ⓢ1, ⓢ2	•ⓢ1 ≦ ⓢ2 时导通状态 •ⓢ1 > ⓢ2 时非导通状态		
	AND<=	ⓢ1, ⓢ2			
	OR<=	ⓢ1, ⓢ2			
	LD<	ⓢ1, ⓢ2	•ⓢ1 < ⓢ2 时导通状态 •ⓢ1 ≧ ⓢ2 时非导通状态		
	AND<	ⓢ1, ⓢ2			
	OR<	ⓢ1, ⓢ2			
	LD>=	ⓢ1, ⓢ2	•ⓢ1 ≧ ⓢ2 时导通状态 •ⓢ1 < ⓢ2 时非导通状态		
	AND>=	ⓢ1, ⓢ2			
	OR>=	ⓢ1, ⓢ2			
	LD>	ⓢ1, ⓢ2	•ⓢ1 > ⓢ2 时导通状态 •ⓢ1 ≦ ⓢ2 时非导通状态		
	AND>	ⓢ1, ⓢ2			
	OR>	ⓢ1, ⓢ2			

表 2.10 比较运算指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
32 位数据比较	LDD=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)=(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≠(ⓢ2+1, ⓢ2) 时非导通状态		6-4
	ANDD=	ⓢ1, ⓢ2			
	ORD=	ⓢ1, ⓢ2			
	LDD<>	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≠(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)=(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDD<>	ⓢ1, ⓢ2			
	ORD<>	ⓢ1, ⓢ2			
	LDD<=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≦(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)＞(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDD<=	ⓢ1, ⓢ2			
	ORD<=	ⓢ1, ⓢ2			
	LDD<	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)＜(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≧(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDD<	ⓢ1, ⓢ2			
	ORD<	ⓢ1, ⓢ2			
	LDD>=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≧(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)＜(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDD>=	ⓢ1, ⓢ2			
	ORD>=	ⓢ1, ⓢ2			
	LDD>	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)＞(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≦(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDD>	ⓢ1, ⓢ2			
	ORD>	ⓢ1, ⓢ2			
浮动小数点数据比较（单精度）	LDE=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)=(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≠(ⓢ2+1, ⓢ2) 时非导通状态		6-6
	ANDE=	ⓢ1, ⓢ2			
	ORE=	ⓢ1, ⓢ2			
	LDE<>	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≠(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)=(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDE<>	ⓢ1, ⓢ2			
	ORE<>	ⓢ1, ⓢ2			
	LDE<=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≦(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)＞(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDE<=	ⓢ1, ⓢ2			
	ORE<=	ⓢ1, ⓢ2			
	LDE<	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)＜(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≧(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDE<	ⓢ1, ⓢ2			
	ORE<	ⓢ1, ⓢ2			
	LDE>=	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)≧(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)＜(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDE>=	ⓢ1, ⓢ2			
	ORE>=	ⓢ1, ⓢ2			
	LDE>	ⓢ1, ⓢ2	• (ⓢ1+1, ⓢ1)＞(ⓢ2+1, ⓢ2) 时导通状态 • (ⓢ1+1, ⓢ1)≦(ⓢ2+1, ⓢ2) 时非导通状态		
	ANDE>	ⓢ1, ⓢ2			
	ORE>	ⓢ1, ⓢ2			

表 2.10 比较运算指令（续）

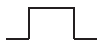
分类	指令名	自变量	处理内容	执行条件	说明页面
浮动小数点数据比较（双精度）	LDED=	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) = (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≠ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		6-9
	ANDED=	ⓢ1, ⓓ2			
	ORED=	ⓢ1, ⓓ2			
	LDED<>	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≠ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) = (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		
	ANDED<>	ⓢ1, ⓓ2			
	ORED<>	ⓢ1, ⓓ2			
	LDED<=	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≦ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) > (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		
	ANDED<=	ⓢ1, ⓓ2			
	ORED<=	ⓢ1, ⓓ2			
	LDED<	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) < (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≧ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		
	ANDED<	ⓢ1, ⓓ2			
	ORED<	ⓢ1, ⓓ2			
	LDED>=	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≧ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) < (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		
	ANDED>=	ⓢ1, ⓓ2			
	ORED>=	ⓢ1, ⓓ2			
	LDED>	ⓢ1, ⓓ2	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) > (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时导通状态 • (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1) ≦ (ⓓ2+3, ⓓ2+2, ⓓ2+1, ⓓ2) 时非导通状态		
	ANDED>	ⓢ1, ⓓ2			
	ORED>	ⓢ1, ⓓ2			

表 2.10 比较运算指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
字符串数据比较	LD$=	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)=(㉒的字符串）时导通状态 •(㉑的字符串)≠(㉒的字符串）时非导通状态		6-12
	AND$=	㉑，㉒			
	OR$=	㉑，㉒			
	LD$<>	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)≠(㉒的字符串）时导通状态 •(㉑的字符串)=(㉒的字符串）时非导通状态		
	AND$<>	㉑，㉒			
	OR$<>	㉑，㉒			
	LD$<=	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)≦(㉒的字符串）时导通状态 •(㉑的字符串)＞(㉒的字符串）时非导通状态		
	AND$<=	㉑，㉒			
	OR$<=	㉑，㉒			
	LD$<	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)＜(㉒的字符串）时导通状态 •(㉑的字符串)≧(㉒的字符串）时非导通状态		
	AND$<	㉑，㉒			
	OR$<	㉑，㉒			
	LD$>=	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)≧(㉒的字符串）时导通状态 •(㉑的字符串)＜(㉒的字符串）时非导通状态		
	AND$>=	㉑，㉒			
	OR$>=	㉑，㉒			
	LD$>	㉑，㉒	•将㉑的字符串与㉒的字符串逐个字符进行比较。*2 •(㉑的字符串)＞(㉒的字符串）时导通状态 •(㉑的字符串)≦(㉒的字符串）时非导通状态		
	AND$>	㉑，㉒			
	OR$>	㉑，㉒			

*2: 进行字符串的比较时的比较条件如下所示。
- 一致的条件 ：所有的字符串一致的情况下
- 大字符串的条件 ：不同的字符串的情况下，字符码较大的字符串
  字符串的长度不同的情况下，字符串较长的字符串
- 小字符串的条件 ：不同的字符串的情况下，字符码较小的字符串字
  符串的长度不同的情况下，字符串较短的字符串

表 2.10 比较运算指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
BIN 16 位块数据比较	BKCMP=	ⓢ1, ⓢ2, n, ⓓ	• 将从ⓢ1算起 n 点的数据与从ⓢ2算起 n 点的数据以 BIN 16 位数据进行比较，将比较结果存储到ⓓ中指定的位软元件算起的 n 点中。		6-16
	BKCMP<>	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP<=	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP<	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP>=	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP>	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP=P	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP<>P	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP<=P	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP<P	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP>=P	ⓢ1, ⓢ2, n, ⓓ			
	BKCMP>P	ⓢ1, ⓢ2, n, ⓓ			
BIN 32 位块数据比较	DBKCMP=	ⓢ1, ⓢ2, n, ⓓ	• 将ⓢ1中指定的软元件算起 n 点的 BIN 32 位数据或常数与ⓢ2中指定的软元件算起 n 点的 BIN 32 位数据进行比较，并将运算结果存储到ⓓ中指定的软元件以后。		6-20
	DBKCMP<>	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP<=	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP<	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP>=	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP>	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP=P	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP<>P	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP<=P	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP<P	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP>=P	ⓢ1, ⓢ2, n, ⓓ			
	DBKCMP>P	ⓢ1, ⓢ2, n, ⓓ			

2.4.2 算术运算指令

表 2.11 算术运算指令

分类	指令名	自变量	处理内容	执行条件	说明页面
BIN 16 位加减法	+	ⓢ1, ⓢ2, ⓓ	•ⓢ1+ⓢ2→ⓓ		6-24
	-	ⓢ1, ⓢ2, ⓓ	•ⓢ1-ⓢ2→ⓓ		
BIN 32 位加减法	D+	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1)+(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		6-26
	D-	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1)-(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		
BIN 16 位乘除法	*	ⓢ1, ⓢ2, ⓓ	•ⓢ1 × ⓢ2→(ⓓ+1, ⓓ)		6-28
	/	ⓢ1, ⓢ2, ⓓ	•ⓢ1 ÷ ⓢ2→商ⓓ, 余 (ⓓ+1)		
BIN 32 位乘除法	D*	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1) × (ⓢ2+1, ⓢ2) →(ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		6-30
	D/	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1) ÷ (ⓢ2+1, ⓢ2) →商 (ⓓ+1, ⓓ), 余 (ⓓ+3, ⓓ+2)		
BCD 4 位加减法	B+	ⓢ1, ⓢ2, ⓓ	•ⓢ1+ⓢ2→ⓓ		6-32
	B-	ⓢ1, ⓢ2, ⓓ	•ⓢ1-ⓢ2→ⓓ		
BCD 8 位加减法	DB+	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1)+(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		6-34
	DB-	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1)-(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		
BCD 4 位乘除法	B 仏	ⓢ1, ⓢ2, ⓓ	•ⓢ1 × ⓢ2→(ⓓ+1, ⓓ)		6-36
	B/	ⓢ1, ⓢ2, ⓓ	•ⓢ1 ÷ ⓢ2→商ⓓ, 余 (ⓓ+1)		
BCD 8 位乘除法	DB 仏	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1) × (ⓢ2+1, ⓢ2) →(ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		6-38
	DB/	ⓢ1, ⓢ2, ⓓ	•(ⓢ1+1, ⓢ1) ÷ (ⓢ2+1, ⓢ2) →商 (ⓓ+1, ⓓ), 余 (ⓓ+3, ⓓ+2)		

表 2.11 算术运算指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
浮动小数点数据加减法（单精度）	E+	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1)+(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		6-41
	E-	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1)-(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		
浮动小数点数据加减法（双精度）	ED+	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1)+ (ⓢ2+3, ⓢ2+2, ⓢ2+1, ⓢ2)→ (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		6-44
	ED-	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1)− (ⓢ2+3, ⓢ2+2, ⓢ2+1, ⓢ2)→ (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		
浮动小数点数据乘除法（单精度）	E*	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1)×(ⓢ2+1, ⓢ2)→(ⓓ+1, ⓓ)		6-46
	E/	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1)÷(ⓢ2+1, ⓢ2)→商 (ⓓ+1, ⓓ)		
浮动小数点数据乘除法（双精度）	ED*	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1)× (ⓢ2+3, ⓢ2+2, ⓢ2+1, ⓢ2)→ (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		6-48
	ED/	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+3, ⓢ1+2, ⓢ1+1, ⓢ1)÷ (ⓢ2+3, ⓢ2+2, ⓢ2+1, ⓢ2)→ 商 (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ)		
BIN 16 位数据块加减法	BK+	ⓢ1, ⓢ2, n, ⓓ	• 将从ⓢ1算起 n 点的 BIN 16 位数据与从ⓢ2算起 n 点的数据进行批量加法运算。		6-50
	BK-	ⓢ1, ⓢ2, n, ⓓ	• 将从ⓢ1算起 n 点的 BIN 16 位数据与从ⓢ2算起 n 点的数据进行批量减法运算。		
BIN 32 位数据块加减法	DBK+	ⓢ1, ⓢ2, n, ⓓ	• 将ⓢ1中指定的软元件算起 n 点的 BIN 32 位数据或常数与ⓢ2中指定的软元件算起 n 点的 BIN 32 位数据进行加法运算，并将运算结果存储到ⓓ中指定的软元件以后。		6-53
	DBK-	ⓢ1, ⓢ2, n, ⓓ	• 将ⓢ1中指定的软元件算起 n 点的 BIN 32 位数据或常数与ⓢ2中指定的软元件算起 n 点的 BIN 32 位数据进行减法运算，并将运算结果存储到ⓓ中指定的软元件以后。		
字符串数据合并	$+	ⓢ1, ⓢ2, ⓓ	• 将ⓢ1中指定的字符串与ⓢ2中指定的字符串合并后，存储到ⓓ以后。		6-57
BIN 数据递增	INC	ⓓ	• ⓓ+1→ⓓ		6-59
	DINC	ⓓ	• (ⓓ+1, ⓓ)+1→(ⓓ+1, ⓓ)		6-61
	DEC	ⓓ	• ⓓ-1→ⓓ		6-59
	DDEC	ⓓ	• (ⓓ+1, ⓓ)-1→(ⓓ+1, ⓓ)		6-61

2.4.3 数据转换指令

表 2.12 数据转换指令

分类	指令名	自变量	处理内容	执行条件	说明页面
BCD 转换	BCD	Ⓢ，ⓓ	• Ⓢ —BCD转换→ ⓓ BIN(0～9999)		6-63
	DBCD	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —BCD转换→ (ⓓ+1, ⓓ) BIN(0～99999999)		
BIN 转换	BIN	Ⓢ，ⓓ	• Ⓢ —BIN转换→ ⓓ BCD(0～9999)		6-66
	DBIN	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —BIN转换→ (ⓓ+1, ⓓ) BCD(0～99999999)		
BIN ↓ 浮动小数点转换（单精度）	FLT	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为实数→ ⓓ BIN(－32768～32767)		6-69
	DFLT	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为实数→ (ⓓ+1, ⓓ) BIN(-2147483648～2147483647)		
BIN ↓ 浮动小数点转换（双精度）	FLTD	Ⓢ，ⓓ	• Ⓢ —转换为实数→ (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ) BIN(－32768～32767)		6-72
	DFLTD	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为实数→ (ⓓ+3, ⓓ+2, ⓓ+1, ⓓ) BIN(-2147483648～2147483647)		
浮动小数点（单精度） ↓ BIN 转换	INT	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为BIN→ ⓓ 实数(－32768～32767)		6-74
	DINT	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为BIN→ (ⓓ+1, ⓓ) 实数(-2147483648～2147483647)		
浮动小数点（双精度） ↓ BIN 转换	INTD	Ⓢ，ⓓ	• (Ⓢ+3, Ⓢ+2, Ⓢ+1, Ⓢ) —转换为BIN→ ⓓ 实数(－32768～32767)		6-77
	DINTD	Ⓢ，ⓓ	• (Ⓢ+3, Ⓢ+2, Ⓢ+1, Ⓢ) —转换为BIN→ (ⓓ+1, ⓓ) 实数(-2147483648～2147483647)		
BIN 16 位 ↓ 32 位转换	DBL	Ⓢ，ⓓ	• Ⓢ —转换→ (ⓓ+1, ⓓ) BIN(-32768～32767)		6-80
	WORD	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换→ ⓓ BIN(-32768～32767)		6-82
BIN ↓ 格雷码转换	GRY	Ⓢ，ⓓ	• Ⓢ —转换为格雷码→ ⓓ BIN(-32768～32767)		6-84
	DGRY	Ⓢ，ⓓ	• (Ⓢ+1, Ⓢ) —转换为格雷码→ (ⓓ+1, ⓓ) BIN(-2147483648～2147483647)		

表 2.12 数据转换指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
格雷码 ↓ BIN 转换	GBIN	Ⓢ，ⓓ	• Ⓢ 转换为BIN数据 → ⓓ 格雷码(-32768～32767)		6-86
	DGBIN	Ⓢ，ⓓ	• (Ⓢ+1,Ⓢ) 转换为BIN数据 → (ⓓ+1,ⓓ) 格雷码(-2147483648～2147483647)		
2 的补数	NEG	ⓓ	• ⓓ → ⓓ BIN数据		6-88
	DNEG	ⓓ	• (ⓓ+1,ⓓ) → (ⓓ+1,ⓓ) BIN数据		
	ENEG	ⓓ	• (ⓓ+1,ⓓ) → (ⓓ+1,ⓓ) 实数数据		6-90
	EDNEG	ⓓ	• (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) → (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) 实数数据		6-92
块转换	BKBCD	Ⓢ，n，ⓓ	•将从Ⓢ算起 n 点的 BIN 数据批量地转换为 BCD 数据后，存储到ⓓ以后。		6-94
	BKBIN	Ⓢ，n，ⓓ	•将从Ⓢ算起 n 点的 BCD 数据批量地转换为 BIN 数据后，存储到ⓓ以后。		6-97
浮动小数点单精度 ↓ 双精度	ECON	Ⓢ，ⓓ	• (Ⓢ+1,Ⓢ) 双精度转换 → (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) 32位浮动小数点型实数		6-100
浮动小数点双精度 ↓ 单精度	EDCON	Ⓢ，ⓓ	• (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) 单精度转换 → (ⓓ+1,ⓓ) 64位浮动小数点型实数		6-102

2.4.4 数据传送指令

表 2.13 数据传送指令

分类	指令名	自变量	处理内容	执行条件	说明页面
16 位数据传送	MOV	ⓢ, ⓓ	• ⓢ → ⓓ		6-104
32 位数据传送	DMOV	ⓢ, ⓓ	• (ⓢ+1,ⓢ) → (ⓓ+1,ⓓ)		
浮动小数点数据传送（单精度）	EMOV	ⓢ, ⓓ	• (ⓢ+1,ⓢ) → (ⓓ+1,ⓓ) 实数数据		6-107
浮动小数点数据传送（双精度）	EDMOV	ⓢ, ⓓ	• (ⓢ+3,ⓢ+2,ⓢ+1,ⓢ) → (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) 实数数据		6-109
字符串数据传送	$MOV	ⓢ, ⓓ	• 将ⓢ中指定的字符串传送至ⓓ中指定的软元件以后。		6-111
16 位数据否定传送	CML	ⓢ, ⓓ	• $\overline{ⓢ}$ → ⓓ		6-114
32 位数据否定传送	DCML	ⓢ, ⓓ	• $\overline{(ⓢ+1,ⓢ)}$ → (ⓓ+1,ⓓ)		
块传送	BMOV	ⓢ, n, ⓓ	ⓢ → ⓓ, n		6-119
同一 16 位数据块传送	FMOV	ⓢ, n, ⓓ	ⓢ → ⓓ, n		6-123
同一 32 位数据块传送	DFMOV	ⓢ, n, ⓓ	ⓢ+1,ⓢ → ⓓ+1,ⓓ, n		6-126
16 位数据替换	XCH	ⓓ1, ⓓ2	• ⓓ1 ↔ ⓓ2		6-129
32 位数据替换	DXCH	ⓓ1, ⓓ2	• (ⓓ1+1,ⓓ1) ↔ (ⓓ2+1,ⓓ2)		
块数据替换	BXCH	n, ⓓ1, ⓓ2	ⓢ ↔ ⓓ, n		6-132
高低字节替换	SWAP	ⓢ	ⓢ b15～b8 b7～b0 [8位][8位] ⓓ b15～b8 b7～b0 [8位][8位]		6-135

2.4.5 程序分支指令

表 2.14 程序分支指令

分类	指令名	自变量	处理内容	执行条件	说明页面
跳转	CJ	ⓟ	• 输入条件成立时跳转至Ⓟⓝ。		6-137
	SCJ	ⓟ	• 从输入条件成立的下一个扫描开始跳转至Ⓟⓝ。		
	JMP	ⓟ	• 无条件跳转至Ⓟⓝ。		
	GOEND	-	• 输入条件成立时跳转至 FEND 指令。		6-141

2.4.6 程序执行控制指令

表 2.15 程序执行控制指令

分类	指令名	自变量	处理内容	执行条件	说明页面
中断禁止	DI	-	• 禁止中断程序的执行。		6-143
中断允许	EI	-	• 解除中断程序的执行禁止。		
中断禁止允许设置	IMASK	Ⓢ	• 对各个中断程序进行中断禁止 / 允许。		
恢复	IRET	-	• 从中断程序返回至顺控程序。		6-152

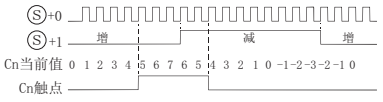
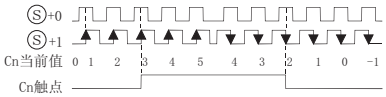
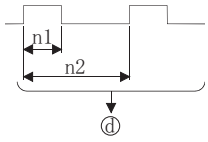
2.4.7 I/O 刷新指令

表 2.16 I/O 刷新指令

分类	指令名	自变量	处理内容	执行条件	说明页面
I/O 刷新	RFS	Ⓢ, n	• 在 1 个扫描的途中对相应的输入输出部分进行刷新。		6-155

2.4.8 其它方便指令

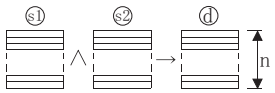
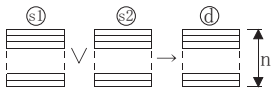
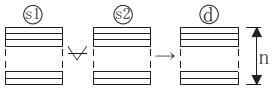
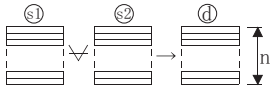
表 2.17 其它方便指令

分类	指令名	自变量	处理内容	执行条件	说明页面
增 / 减计数器	UDCNT1	ⓢ, n, ⓓ			6-157
	UDCNT2	ⓢ, n, ⓓ			6-160
示教定时器	TTMR	n, ⓓ	• (TTMR的ON时间) ×n ⟶ ⓓ n=0:1, n=1:10, n=2:100		6-163
特殊定时器	STMR	ⓢ, n, ⓓ	•根据 STMR 指令输入条件的 ON/OFF，从ⓓ中指定的位软元件算起的 4 点执行以下动作。 •ⓓ+0：OFF 延迟定时器输出 •ⓓ+1：OFF 后单次定时器输出 •ⓓ+2：ON 后单次定时器输出 •ⓓ+3：ON 延迟 +OFF 延迟定时器输出		6-165
就近控制	ROTC	ⓢ, n1, n2, ⓓ	•在被分割为 n1 份的旋转台上，从停止位置开始就近旋转至Ⓢ+1 中指定的位置。		6-168
斜坡信号	RAMP	n1, n2, n3, ⓓ, ⓬	•将ⓓ中指定的软元件数据通过 n3 个扫描由 n1 变为 n2。		6-171
脉冲密度	SPD	ⓢ, n, ⓓ	•将ⓢ中指定的软元件的脉冲输入在 n 中指定的时间进行计数，并存储到ⓓ中指定的软元件中。		6-174
脉冲输出	PLSY	n1, n2, ⓓ	• (n1)Hz ⟶ ⓓ n2次输出		6-176
脉冲宽度调制	PWM	n1, n2, ⓓ			6-178
矩阵输入	MTR	ⓢ, n, ⓓ, ⓬	•从ⓢ中指定的软元件开始依次获取 16 点 n 列的数据，存储到⓬中指定的软元件以后。		6-180

2.5 应用指令

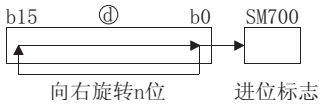
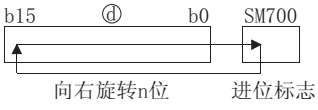
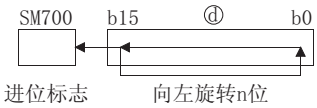
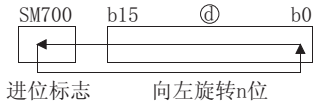
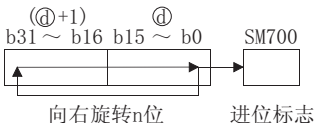
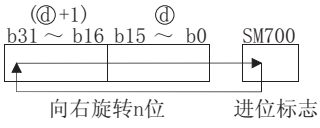
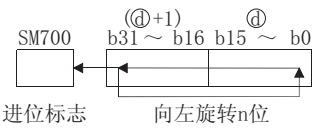
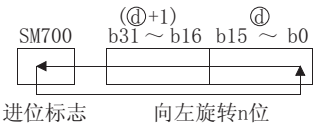
2.5.1 逻辑运算指令

表 2.18 逻辑运算指令

分类	指令名	自变量	处理内容	执行条件	说明页面
逻辑积	WAND	ⓢ1, ⓢ2, ⓓ	• ⓢ1 ∧ ⓢ2 → ⓓ		7-3
	DAND	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1) ∧ (ⓢ2+1, ⓢ2) → (ⓓ+1, ⓓ)		
	BKAND	ⓢ1, ⓢ2, n, ⓓ			7-6
逻辑和	WOR	ⓢ1, ⓢ2, ⓓ	• ⓢ1 ∨ ⓢ2 → ⓓ		7-9
	DOR	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1) ∨ (ⓢ2+1, ⓢ2) → (ⓓ+1, ⓓ)		
	BKOR	ⓢ1, ⓢ2, n, ⓓ			7-12
排他逻辑和	WXOR	ⓢ1, ⓢ2, ⓓ	• ⓢ1 ⊻ ⓢ2 → ⓓ		7-15
	DXOR	ⓢ1, ⓢ2, ⓓ	• (ⓢ1+1, ⓢ1) ⊻ (ⓢ2+1, ⓢ2) → (ⓓ+1, ⓓ)		
	BKXOR	ⓢ1, ⓢ2, n, ⓓ			7-18
否定排他逻辑和	WXNR	ⓢ1, ⓢ2, ⓓ	• $\overline{ⓢ1 ⊻ ⓢ2}$ → ⓓ		7-21
	DXNR	ⓢ1, ⓢ2, ⓓ	• $\overline{(ⓢ1+1, ⓢ1) ⊻ (ⓢ2+1, ⓢ2)}$ → (ⓓ+1, ⓓ)		
	BKXNR	ⓢ1, ⓢ2, n, ⓓ			7-24

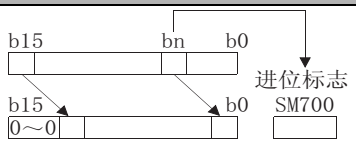
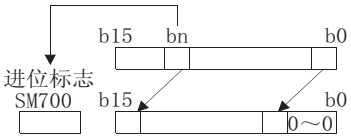
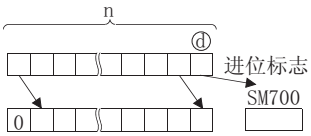
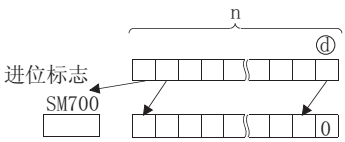
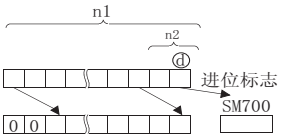
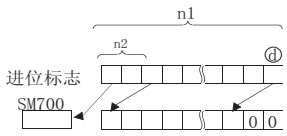
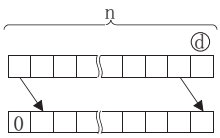
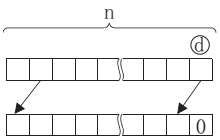
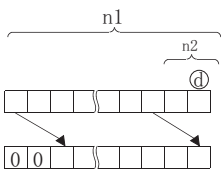
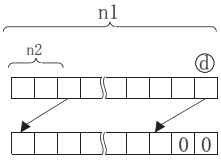
2.5.2 旋转指令

表 2.19 旋转指令

分类	指令名	自变量	处理内容	执行条件	说明页面
右旋转	ROR	n, ⓓ			7-27
	RCR	n, ⓓ			
左旋转	ROL	n, ⓓ			7-31
	RCL	n, ⓓ			
右旋转	DROR	n, ⓓ			7-35
	DRCR	n, ⓓ			
左旋转	DROL	n, ⓓ			7-38
	DRCL	n, ⓓ			

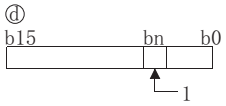
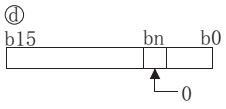
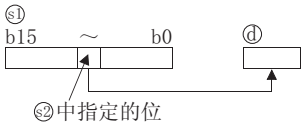
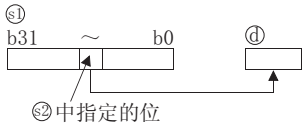
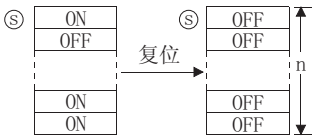
2.5.3 移动指令

表 2.20 移动指令

分类	指令名	自变量	处理内容	执行条件	说明页面
16 位数据的 n 位移动	SFR	n, ⓓ			7-41
	SFL	n, ⓓ			
n 位数据的 1 位移动	BSFR	n, ⓓ			7-44
	BSFL	n, ⓓ			
n 位数据的 n 位移动	SFTBR	n1, n2, ⓓ			7-47
	SFTBL	n1, n2, ⓓ			
n 字数据的 1 字移动	DSFR	n, ⓓ			7-50
	DSFL	n, ⓓ			
n 字数据的 n 字移动	SFTWR	n1, n2, ⓓ			7-53
	SFTWL	n1, n2, ⓓ			

2.5.4 位处理指令

表 2.21 位处理指令

分类	指令名	自变量	处理内容	执行条件	说明页面
位设置 / 复位	BSET	n, ⓓ			7-56
	BRST	n, ⓓ			
位测试	TEST	ⓢ1, ⓢ2, ⓓ			7-59
	DTEST	ⓢ1, ⓢ2, ⓓ			
位软元件批量复位	BKRST	ⓢ, n			7-63

2.5.5 数据处理指令

表 2.22 数据处理指令

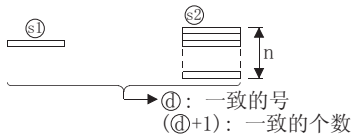
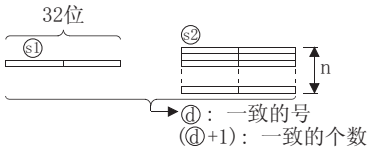
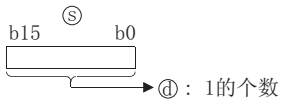
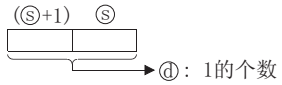
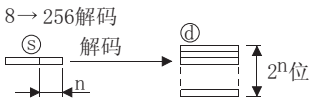
分类	指令名	自变量	处理内容	执行条件	说明页面
数据查找	SER	ⓢ1, ⓢ2, n, ⓓ			7-65
	DSER	ⓢ1, ⓢ2, n, ⓓ			
位校验	SUM	Ⓢ, ⓓ			7-69
	DSUM	Ⓢ, ⓓ			
解码	DECO	Ⓢ, n, ⓓ			7-72
编码	ENCO	Ⓢ, n, ⓓ			7-74
7 段解码	SEG	Ⓢ, ⓓ			7-76
分离・合并	DIS	Ⓢ, n, ⓓ	・将Ⓢ中指定的 16 位数据以 4 位为单位进行分离后，存储到从ⓓ算起 n 点的低位 4 位中。(n ≦ 4)		7-79
	UNI	Ⓢ, n, ⓓ	・将Ⓢ中指定的软元件算起 n 点的低位 4 位数据进行合并后，存储到ⓓ中指定的软元件中。(n ≦ 4)		7-81
	NDIS	ⓢ1, ⓢ2, ⓓ	・将ⓢ1中指定的软元件以后的数据分离为ⓢ2以后中指定的位后，从ⓓ中指定的软元件开始按顺序存储。		7-83
	NUNI	ⓢ1, ⓢ2, ⓓ	・将ⓢ1中指定的软元件以后的数据合并为ⓢ2以后中指定的各个位后，从ⓓ中指定的软元件开始按顺序存储。		
	WTOB	Ⓢ, n, ⓓ	・将Ⓢ中指定的软元件算起 n 点 16 位数据以 8 位为单位进行分解后，从ⓓ中指定的软元件开始按顺序存储。		7-88
	BTOW	Ⓢ, n, ⓓ	・将Ⓢ中指定的软元件算起 n 点的 16 位数据的低位 8 位合并为 16 位后，从ⓓ中指定的软元件开始按顺序存储。		

表 2.22　数据处理指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
查找	MAX	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点的数据以 16 位为单位进行查找，将最大值存储到ⓓ中指定的软元件中。		7-93
	DMAX	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 2 × n 点的数据以 32 位为单位进行查找，将最大值存储到ⓓ中指定的软元件中。		
	MIN	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点的数据以 16 位为单位进行查找，将最小值存储到ⓓ中指定的软元件中。		7-96
	DMIN	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 2 × n 点的数据以 32 位为单位进行查找，将最小值存储到ⓓ中指定的软元件中。		
排序	SORT	ⓢ1，n，ⓢ2，ⓓ1，ⓓ2	•将ⓢ1中指定的软元件算起 n 点的数据以 16 位为单位进行排序。〔需要 n × (n-1)/2 个扫描〕		7-99
	DSORT	ⓢ1，n，ⓢ2，ⓓ1，ⓓ2	•将ⓢ1中指定的软元件算起 2 × n 点的数据以 32 位为单位进行排序。〔需要 n × (n-1)/2 个扫描〕		
合计值计算	WSUM	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点的 16 位 BIN 数据全部进行加法运算后，存储到ⓓ中指定的软元件中。		7-104
	DWSUM	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点的 32 位 BIN 数据全部进行加法运算后，存储到ⓓ中指定的软元件中。		7-106
平均值计算	MEAN	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点（16 位单位）的平均值进行计算后，存储到ⓓ中指定的软元件中。		7-108
	DMEAN	ⓢ，n，ⓓ	•将ⓢ中指定的软元件算起 n 点（32 位单位）的平均值进行计算后，存储到ⓓ中指定的软元件中。		

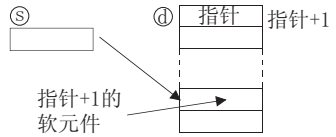
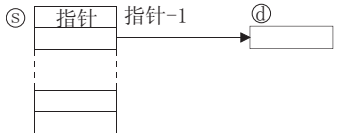
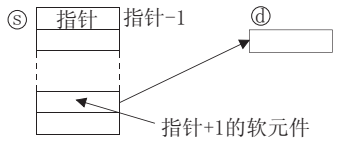
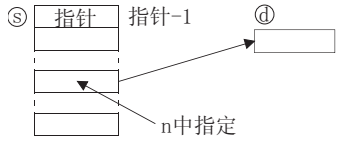
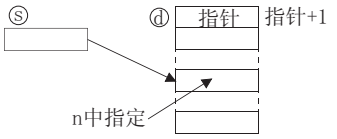
2.5.6 结构体指令

表 2.23 结构体指令

分类	指令名	自变量	处理内容	执行条件	说明页面
循环	FOR	n	• 在 FOR ～ NEXT 之间执行 n 次。		7-111
	NEXT	-			
	BREAK	ⓟ, ⓓ	• 将 FOR ～ NEXT 之间的执行强制结束后，跳转至指针ⓟ。		7-114
子程序调用	CALL	ⓟ	• 输入条件成立时执行ⓟ子程序。		7-116
	RET	-	• 从子程序中返回。		7-118
选择刷新	COM	-	• 进行智能功能模块的自动刷新、一般数据及 CPU 共享存储器的自动刷新。		7-119
	CCOM	-	• 输入条件成立时进行智能功能模块的自动刷新、CPU 共享存储器的自动刷新及通信处理。		7-127

2.5.7 数据表操作指令

表 2.24 数据表操作指令

分类	指令名	自变量	处理内容	执行条件	说明页面
数据表处理	FIFW	ⓢ, ⓓ			7-128
	FIFR	ⓢ, ⓓ			7-131
	FPOP	ⓢ, ⓓ			7-134
	FDEL	ⓢ, n, ⓓ			7-137
	FINS	ⓢ, n, ⓓ			

2.5.8 缓冲存储器访问指令

表 2.25 缓冲存储器访问指令

分类	指令名	自变量	处理内容	执行条件	说明页面
数据读取	FROM	n1, n2, n3, ⓓ	• 从智能功能模块以 16 位为单位进行数据读取。		7-140
	DFRO	Ⓢ, n1, n2, n3	• 从智能功能模块以 32 位为单位进行数据读取。		
数据写入	TO	Ⓢ, n1, n2, n3	• 将数据以 16 位为单位写入到智能功能模块中。		7-144
	DTO	Ⓢ, n1, n2, n3	• 将数据以 32 位为单位写入到智能功能模块中。		

2.5.9 显示指令

表 2.26 显示指令

分类	指令名	自变量	处理内容	执行条件	说明页面
ASCII 打印	PR	Ⓢ, ⓓ	• SM701 为 OFF 时，将从Ⓢ中指定的软元件算起 8 点（16 字符）的 ASCII 码输出到输出模块中。		7-148
	PR	Ⓢ, ⓓ	• SM701 为 ON 时，将从Ⓢ中指定的软元件开始至 00H 为止的 ASCII 码输出到输出模块中。		
	PRC	Ⓢ, ⓓ	• 将Ⓢ中指定的软元件的注释转换为 ASCII 码后，输出到输出模块中。		7-152
复位	LEDR	-	• 进行报警器的复位以及对 LED 显示器的显示进行复位。		7-156

2.5.10 调试・故障诊断指令

表 2.27 调试・故障诊断指令

分类	指令名	自变量	处理内容	执行条件	说明页面
检查	CHKST	-	• 执行 CHKST 指令时，执行 CHK 指令。 • CHKST 指令非执行时，跳转到 CHK 指令的下一步。		7-159
	CHK	-	• 正常时→SM80：OFF； SD80：0 • 异常时→SM80：ON； SD80：故障 No.		
	CHKCIR	-	• 通过 CHK 指令检查的梯形图模式的变更开始。		7-163
	CHKEND	-	• 通过 CHK 指令检查的梯形图模式的变更结束。		

2.5.11 字符串处理指令

表 2.28 字符串处理指令

分类	指令名	自变量	处理内容	执行条件	说明页面
BIN ↓ 10 进制 ASCII	BINDA	Ⓢ，ⓓ	•将Ⓢ中指定的 1 字 BIN 值转换为 5 位数 10 进制 ASCII 值后，存储到ⓓ中指定的字软元件中。		7-167
	DBINDA	Ⓢ，ⓓ	•将Ⓢ中指定的 2 字 BIN 值转换为 10 位数 10 进制 ASCII 值后，存储到ⓓ中指定的字软元件编号以后。		
BIN ↓ 16 进制 ASCII	BINHA	Ⓢ，ⓓ	•将Ⓢ中指定的 1 字 BIN 值转换为 4 位数 16 进制 ASCII 值后，存储到ⓓ中指定的字软元件编号以后。		7-171
	DBINHA	Ⓢ，ⓓ	•将Ⓢ中指定的 2 字 BIN 值转换为 8 位数 16 进制 ASCII 值后，存储到ⓓ中指定的字软元件编号以后。		
BCD ↓ 10 进制 ASCII	BCDDA	Ⓢ，ⓓ	•将Ⓢ中指定的 1 字 BCD 值转换为 4 位数 10 进制 ASCII 值后，存储到ⓓ中指定的字软元件编号以后。		7-175
	DBCDDA	Ⓢ，ⓓ	•将Ⓢ中指定的 2 字 BCD 值转换为 8 位数 10 进制 ASCII 值后，存储到ⓓ中指定的字软元件编号以后。		
10 进制 ASCII ↓ BIN	DABIN	Ⓢ，ⓓ	•将Ⓢ中指定的 5 位数 10 进制 ASCII 值转换为 1 字 BIN 值后，存储到ⓓ中指定的字软元件编号以后。		7-179
	DDABIN	Ⓢ，ⓓ	•将Ⓢ中指定的 10 位数 10 进制 ASCII 值转换为 2 字 BIN 值后，存储到ⓓ中指定的字软元件编号以后。		
16 进制 ASCII ↓ BIN	HABIN	Ⓢ，ⓓ	•将Ⓢ中指定的 4 位数 16 进制 ASCII 值转换为 1 字 BIN 值后，存储到ⓓ中指定的字软元件编号以后。		7-182
	DHABIN	Ⓢ，ⓓ	•将Ⓢ中指定的 8 位数 16 进制 ASCII 值转换为 2 字 BIN 值后，存储到ⓓ中指定的字软元件编号以后。		
10 进制 ASCII ↓ BCD	DABCD	Ⓢ，ⓓ	•将Ⓢ中指定的 4 位数 10 进制 ASCII 值转换为 1 字 BCD 值后，存储到ⓓ中指定的字软元件编号以后。		7-185
	DDABCD	Ⓢ，ⓓ	•将Ⓢ中指定的 8 位数 10 进制 ASCII 值转换为 2 字 BCD 值后，存储到ⓓ中指定的字软元件编号以后。		
软元件注释的读取	COMRD	Ⓢ，ⓓ	•将Ⓢ中指定的软元件的注释数据存储到ⓓ中指定的软元件中。		7-188
字符串的长度检测	LEN	Ⓢ，ⓓ	•将Ⓢ中指定的软元件中存储的字符串数据的长度（字符数）存储到ⓓ中指定的软元件中。		7-191

表 2.28　字符串处理指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
BIN ↓ 10 进制字符串	STR	ⓢ1，ⓢ2，ⓓ	•将ⓢ2中指定的 1 字 BIN 值，转换为ⓢ1中指定的全部位数及小数部分位数的 10 进制字符串后，存储到ⓓ中指定的软元件中。		7-193
	DSTR	ⓢ1，ⓢ2，ⓓ	•将ⓢ2中指定的 2 字 BIN 值，转换为ⓢ1中指定的全部位数及小数部分位数的 10 进制字符串后，存储到ⓓ中指定的软元件中。		
10 进制字符串 ↓ BIN	VAL	Ⓢ，ⓓ1，ⓦ2	•将Ⓢ中指定的包含了小数点的字符串转换为 1 字 BIN 值及小数部分位数后，存储到ⓓ1、ⓦ2中指定的软元件中。		7-199
	DVAL	Ⓢ，ⓓ1，ⓦ2	•将Ⓢ中指定的包含了小数点的字符串转换为 2 字 BIN 值及小数部分位数后，存储到ⓓ1、ⓦ2中指定的软元件中。		
浮动小数点 ↓ 字符串	ESTR	ⓢ1，ⓢ2，ⓓ	•将ⓢ1中指定的浮动小数点数据转换为字符串后，存储到ⓓ中指定的软元件中。		7-204
字符串 ↓ 浮动小数点	EVAL	Ⓢ，ⓓ	•将Ⓢ中指定的字符串转换为浮动小数点数据后，存储到ⓓ中指定的软元件中。		7-210
16 进制 BIN ↓ ASCII	ASC	Ⓢ，n，ⓓ	•将Ⓢ中指定的软元件编号以后的 1 字 BIN 值转换为 16 进制 ASCII 后，以 n 中指定的字符数存储到ⓓ中指定的字软元件编号以后。		7-214
ASCII ↓ 16 进制 BIN	HEX	Ⓢ，n，ⓓ	•将Ⓢ中指定的字软元件以后的 16 进制 ASCII 数据，以 n 中指定的字符数转换为 BIN 值后，存储到ⓓ中指定的软元件编号以后。		7-217
字符串处理	RIGHT	Ⓢ，n，ⓓ	•将Ⓢ中指定的字符串的最终字符算起的 n 个字符，存储到ⓓ中指定的软元件中。		7-220
	LEFT	Ⓢ，n，ⓓ	•将Ⓢ中指定的字符串的起始字符算起的 n 个字符，存储到ⓓ中指定的软元件中。		
	MIDR	ⓢ1，ⓢ2，ⓓ	•从ⓢ1中指定的字符串的ⓢ2中指定的位置开始将指定的字符数存储到ⓓ中指定的软元件中。		7-223
	MIDW	ⓢ1，ⓢ2，ⓓ	•将ⓢ1的字符串开始将指定的字符数存储到ⓓ的字符串的ⓢ2中指定的位置处。		
	INSTR	ⓢ1，ⓢ2，n，ⓓ	•将ⓢ1的字符串从ⓢ2的字符串的第 n 字符开始进行查找，将一致的位置存储到ⓓ中。		7-227
	STRINS	Ⓢ，n，ⓓ	•将Ⓢ中指定的字符串数据，插入到从ⓓ中指定的字符串数据的起始算起的第 n 个字符（插入位置）处。		7-230
	STRDEL	n1,n2，ⓓ	•从ⓓ中指定的字符串数据的起始算起第 n1 个字符中指定的位置（删除开始位置）开始，删除 n2 个字符。		7-233

表 2.28　字符串处理指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
浮动小数点 ↓ BCD 分解	EMOD	㉃，㉄，ⓓ	•将㉃的浮动小数点数据转换为㉄中指定的小数部分位数的 BCD 后，存储到ⓓ中指定的软元件中。		7-235
BCD ↓ 曷浮动小数点搌	EREXP	㉃，㉄，ⓓ	•将㉃的 BCD 数据以㉄中指定的小数部分位数转换为浮动小数点数据后，存储到ⓓ中指定的软元件中。		7-238

2.5.12 特殊函数指令

表 2.29 特殊函数指令

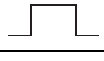
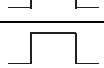
分类	指令名	自变量	处理内容	执行条件	说明页面
三角函数（浮动小数点单精度）	SIN	Ⓢ, ⓓ	• Sin (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-240
	COS	Ⓢ, ⓓ	• Cos (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-244
	TAN	Ⓢ, ⓓ	• Tan (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-248
	ASIN	Ⓢ, ⓓ	• Sin^{-1}(Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-252
	ACOS	Ⓢ, ⓓ	• Cos^{-1}(Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-257
	ATAN	Ⓢ, ⓓ	• Tan^{-1}(Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-262
三角函数（浮动小数点双精度）	SIND	Ⓢ, ⓓ	• Sin (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-242
	COSD	Ⓢ, ⓓ	• Cos (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-246
	TAND	Ⓢ, ⓓ	• Tan (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-250
	ASIND	Ⓢ, ⓓ	• Sin^{-1}(Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-255
	ACOSD	Ⓢ, ⓓ	• Cos^{-1}(Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-260
	ATAND	Ⓢ, ⓓ	• Tan^{-1}(Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-265
度 ↕ 弧度转换	RAD	Ⓢ, ⓓ	• (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ) 度 → 弧度转换		7-267
	RADD	Ⓢ, ⓓ	• (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) 度 → 弧度转换		7-269
	DEG	Ⓢ, ⓓ	• (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ) 弧度 → 度转换		7-271
	DEGD	Ⓢ, ⓓ	• (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ) 弧度 → 度转换		7-273
平方根	SQR	Ⓢ, ⓓ	• $\sqrt{(Ⓢ+1,Ⓢ)}$ ⟶ (ⓓ+1,ⓓ)		7-281
	SQRD	Ⓢ, ⓓ	• $\sqrt{(Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ)}$ ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-283
指数运算	EXP	Ⓢ, ⓓ	• $e^{(Ⓢ+1,Ⓢ)}$ ⟶ (ⓓ+1,ⓓ)		7-285
	EXPD	Ⓢ, ⓓ	• $e^{(Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ)}$ ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-288
自然对数	LOG	Ⓢ, ⓓ	• $\log_e$ (Ⓢ+1,Ⓢ) ⟶ (ⓓ+1,ⓓ)		7-290
	LOGD	Ⓢ, ⓓ	• $\log_e$ (Ⓢ+3,Ⓢ+2,Ⓢ+1,Ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-292

表 2.29　特殊函数指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
幂运算	POW	ⓢ1，ⓢ2，ⓓ	• (ⓢ1+1,ⓢ1)$^{(ⓢ2+1,ⓢ2)}$ ⟶ (ⓓ+1,ⓓ)		7-275
	POWD	ⓢ1，ⓢ2，ⓓ	• ((ⓢ1+3,ⓢ1+2,ⓢ1+1,ⓢ1)$^{(ⓢ2+3,ⓢ2+2,ⓢ2+1,ⓢ2)}$ ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-278
常用对数运算	LOG10	ⓢ，ⓓ	• $\log_{10}$(ⓢ+1,ⓢ) ⟶ (ⓓ+1,ⓓ)		7-294
	LOG10D	ⓢ，ⓓ	• $\log_{10}$(ⓢ+3,ⓢ+2,ⓢ+1,ⓢ) ⟶ (ⓓ+3,ⓓ+2,ⓓ+1,ⓓ)		7-296
随机数发生	RND	ⓓ	• 发生 0 ～ 32767 范围的随机数并存储到ⓓ中指定的软元件中。		7-298
随机数系列变更	SRND	ⓓ	• 根据ⓢ中指定的软元件中存储的 16 位 BIN 数据的内容，对随机数系列进行变更。		
平方根	BSQR	ⓢ，ⓓ	• $\sqrt{ⓢ}$ ⟶ ⓓ+0 整数部分 +1 小数部分		7-300
	BDSQR	ⓢ，ⓓ	• $\sqrt{(ⓢ+1,ⓢ)}$ ⟶ ⓓ+0 整数部分 +1 小数部分		
三角函数	BSIN	ⓢ，ⓓ	• Sinⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-304
	BCOS	ⓢ，ⓓ	• Cosⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-307
	BTAN	ⓢ，ⓓ	• Tanⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-310
	BASIN	ⓢ，ⓓ	• Sin^{-1}ⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-313
	BACOS	ⓢ，ⓓ	• Cos^{-1}ⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-316
	BATAN	ⓢ，ⓓ	• Tan^{-1}ⓢ ⟶ ⓓ+0 符号 +1 整数部分 +2 小数部分		7-319

2.5.13 数据控制指令

表 2.30 数据控制指令

分类	指令名	自变量	处理内容	执行条件	说明页面
上下限极限控制	LIMIT	ⓢ1, ⓢ2, ⓢ3, ⓓ	•ⓢ3＜ⓢ1时 .....将ⓢ1的值存储到ⓓ中。 •ⓢ1≦ⓢ3≦ⓢ2时 .....将ⓢ3的值存储到ⓓ中。 •ⓢ2＜ⓢ3时 .....将ⓢ2的值存储到ⓓ中。		7-321
	DLIMIT	ⓢ1, ⓢ2, ⓢ3, ⓓ	•（ⓢ3+1, ⓢ3）＜（ⓢ1+1, ⓢ1）时 .....将（ⓢ1+1, ⓢ1）的值存储到（ⓓ+1, ⓓ）中。 •（ⓢ1+1, ⓢ1）≦（ⓢ3+1, ⓢ3）＜（ⓢ2+1, ⓢ2）时 .....将（ⓢ3+1, ⓢ3）的值存储到（ⓓ+1, ⓓ）中。 •（ⓢ2, ⓢ2+1）＜（ⓢ3, ⓢ3+1）时 .....将（ⓢ2+1, ⓢ2）的值存储到（ⓓ+1, ⓓ）中。		
死区控制	BAND	ⓢ1, ⓢ2, ⓢ3, ⓓ	•ⓢ1≦ⓢ3≦ⓢ2时..... 0→ⓓ •ⓢ3＜ⓢ1时 ..... ⓢ3-ⓢ1→ⓓ •ⓢ2＜ⓢ3时 ..... ⓢ3-ⓢ2→ⓓ		7-325
	DBAND	ⓢ1, ⓢ2, ⓢ3, ⓓ	•（ⓢ1+1, ⓢ1）≦（ⓢ3+1, ⓢ3）≦（ⓢ2+1, ⓢ2）时 .....0→（ⓓ+1, ⓓ） •（ⓢ3+1, ⓢ3）＜（ⓢ1+1, ⓢ1）时 .....（ⓢ3+1, ⓢ3）-（ⓢ1+1, ⓢ1）→（ⓓ+1, ⓓ） •（ⓢ2+1, ⓢ2）＜（ⓢ3+1, ⓢ3）时 .....（ⓢ3+1, ⓢ3）-（ⓢ2+1, ⓢ2）→（ⓓ+1, ⓓ）		
区域控制	ZONE	ⓢ1, ⓢ2, ⓢ3, ⓓ	•ⓢ3=0 时..... 0→ⓓ •ⓢ3＞0 时..... ⓢ3+ⓢ2→ⓓ •ⓢ3＜0 时..... ⓢ3+ⓢ1→ⓓ		7-329
	DZONE	ⓢ1, ⓢ2, ⓢ3, ⓓ	•（ⓢ3+1, ⓢ3）=0 时 .....0→（ⓓ+1, ⓓ） •（ⓢ3+1, ⓢ3）＞0 时 .....（ⓢ3+1, ⓢ3）+（ⓢ2+1, ⓢ2）→（ⓓ+1, ⓓ） •（ⓢ3+1, ⓢ3）＜0 时 .....（ⓢ3+1, ⓢ3）+（ⓢ1+1, ⓢ1）→（ⓓ+1, ⓓ）		

表 2.31 数据控制指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
点坐标数据	SCL	ⓢ1, ⓢ2, ⓓ	• 对于ⓢ2中指定的标度用转换数据（16 位数据单位），通过ⓢ1中指定的输入值进行标度，并将运算结果存储到ⓓ中指定的软元件中。 标度转换是基于ⓢ2中指定的软元件以后存储的标度用转换数据进行的。		7-333
	DSCL	ⓢ1, ⓢ2, ⓓ	• 对于ⓢ2中指定的标度用转换数据（32 位数据单位），通过ⓢ1中指定的输入值进行标度，并将运算结果存储到ⓓ中指定的软元件中。 标度转换是基于ⓢ2中指定的软元件以后存储的标度用转换数据进行的。		
X/Y 坐标数据	SCL2	ⓢ1, ⓢ2, ⓓ	• 对于ⓢ2中指定的标度用转换数据（16 位数据单位），通过ⓢ1中指定的输入值进行标度，并将运算结果存储到ⓓ中指定的软元件中。 标度转换是基于ⓢ2中指定的软元件以后存储的标度用转换数据进行的。		7-337
	DSCL2	ⓢ1, ⓢ2, ⓓ	• 对于ⓢ2中指定的标度用转换数据（32 位数据单位），通过ⓢ1中指定的输入值进行标度，并将运算结果存储到ⓓ中指定的软元件中。 标度转换是基于ⓢ2中指定的软元件以后存储的标度用转换数据进行的。		

2.5.14 切换指令

表 2.32 切换指令

分类	指令名	自变量	处理内容	执行条件	说明页面
块号切换	RSET	ⓢ	• 将扩展文件寄存器的块号变更为ⓢ中指定的编号。		7-340
文件设置	QDRSET	ⓢ	• 对作为文件寄存器使用的文件名进行设置。		7-343
	QCDSET	ⓢ	• 对作为注释文件使用的文件名进行设置。		7-346

2.5.15 时钟用指令

表 2.33 时钟用指令

分类	指令名	自变量	处理内容	执行条件	说明页面
时钟数据的读取 / 写入	DATERD	ⓓ	• (时钟因子) → ⓓ +0 年 +1 月 +2 日 +3 时 +4 分 +5 秒 +6 星期		7-349
	DATEWR	ⓢ	• ⓓ +0 年 → (时钟因子) +1 月 +2 日 +3 时 +4 分 +5 秒 +6 星期		7-352
时钟数据的加减法运算	DATE+	ⓢ1, ⓢ2, ⓓ	ⓢ1 时 分 秒 + ⓢ2 时 分 秒 → ⓓ 时 分 秒		7-355
	DATE-	ⓢ1, ⓢ2, ⓓ	ⓢ1 时 分 秒 − ⓢ2 时 分 秒 → ⓓ 时 分 秒		7-357
时钟数据的转换	SECOND	Ⓢ, ⓓ	Ⓢ 时 分 秒 → ⓓ 秒(低位) 秒(高位)		7-360
	HOUR	Ⓢ, ⓓ	Ⓢ 秒(低位) 秒(高位) → ⓓ 时 分 秒		7-362

表 2.34 时钟用指令（续）

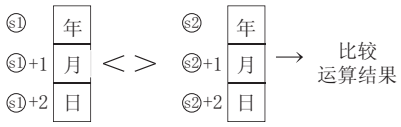
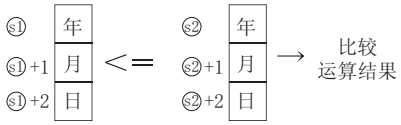
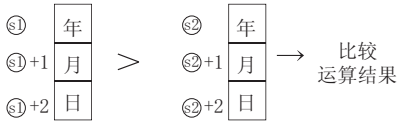
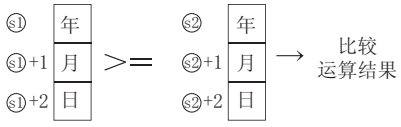
分类	指令名	自变量	处理内容	执行条件	说明页面
日期比较	LDDT=	ⓢ1, ⓢ2, n			7-364
	ANDDT=	ⓢ1, ⓢ2, n			
	ORDT=	ⓢ1, ⓢ2, n			
	LDDT<>	ⓢ1, ⓢ2, n			
	ANDDT<>	ⓢ1, ⓢ2, n			
	ORDT<>	ⓢ1, ⓢ2, n			
	LDDT<	ⓢ1, ⓢ2, n			
	ANDDT<	ⓢ1, ⓢ2, n			
	ORDT<	ⓢ1, ⓢ2, n			
	LDDT<=	ⓢ1, ⓢ2, n			
	ANDDT<=	ⓢ1, ⓢ2, n			
	ORDT<=	ⓢ1, ⓢ2, n			
	LDDT>	ⓢ1, ⓢ2, n			
	ANDDT>	ⓢ1, ⓢ2, n			
	ORDT>	ⓢ1, ⓢ2, n			
	LDDT>=	ⓢ1, ⓢ2, n			
	ANDDT>=	ⓢ1, ⓢ2, n			
	ORDT>=	ⓢ1, ⓢ2, n			

表 2.35 时钟用指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
时钟比较	LDTM=	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 = ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		7-369
	ANDTM=	ⓢ1, ⓢ2, n			
	ORTM=	ⓢ1, ⓢ2, n			
	LDTM<>	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 <> ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		
	ANDTM<>	ⓢ1, ⓢ2, n			
	ORTM<>	ⓢ1, ⓢ2, n			
	LDTM<	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 < ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		
	ANDTM<	ⓢ1, ⓢ2, n			
	ORTM<	ⓢ1, ⓢ2, n			
	LDTM<=	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 <= ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		
	ANDTM<=	ⓢ1, ⓢ2, n			
	ORTM<=	ⓢ1, ⓢ2, n			
	LDTM>	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 > ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		
	ANDTM>	ⓢ1, ⓢ2, n			
	ORTM>	ⓢ1, ⓢ2, n			
	LDTM>=	ⓢ1, ⓢ2, n	ⓢ1 时 / ⓢ1+1 分 / ⓢ1+2 秒 >= ⓢ2 时 / ⓢ2+1 分 / ⓢ2+2 秒 → 比较运算结果		
	ANDTM>=	ⓢ1, ⓢ2, n			
	ORTM>=	ⓢ1, ⓢ2, n			

2.5.16 扩展时钟用指令

表 2.36 扩展时钟用指令

分类	指令名	自变量	处理内容	执行条件	说明页面
扩展时钟数据的读取	S_DATERD	ⓓ	• (时钟因子) →(D)+0 年 +1 月 +2 日 +3 时 +4 分 +5 秒 +6 星期 +7 1/1000秒		7-374
扩展时钟数据的加减法运算	S_DATE+	ⓢ1, ⓢ2, ⓓ	(S1) 时 分 秒 — 1/1000秒 + (S2) 时 分 秒 — 1/1000秒 → (D) 时 分 秒 — 1/1000秒		7-377
	S_DATE-	ⓢ1, ⓢ2, ⓓ	(S1) 时 分 秒 — 1/1000秒 − (S2) 时 分 秒 — 1/1000秒 → (D) 时 分 秒 — 1/1000秒		7-380

2.5.17 程序控制用指令

表 2.37 程序控制用指令

分类	指令名	自变量	处理内容	执行条件	说明页面
程序控制用指令	PSTOP	ⓢ	• 将指定的程序置为待机类型。		7-383
	POFF	ⓢ	• 将指定程序的 OUT 指令的线圈置 OFF 后，置为待机类型。		7-385
	PSCAN	ⓢ	• 将指定的程序登录为扫描执行类型。		7-387
	PLOW	ⓢ	• 将指定的程序登录为低速执行类型。		7-389
	LDPCHK	ⓢ	• 指定文件名程序为执行过程中时置为导通状态。 • 指定文件名程序为非执行时置为非导通状态。		7-391
	ANDPCHK	ⓢ			
	ORPCHK	ⓢ			

2.5.18 其它指令

表 2.38 其它指令

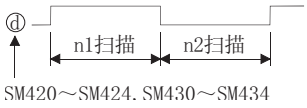
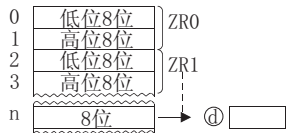
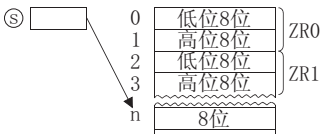
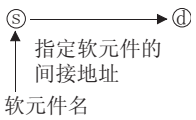
分类	指令名	自变量	处理内容	执行条件	说明页面
WDT 复位	WDT	-	• 在顺控程序中对 WDT 进行复位。		7-393
定时时钟	DUTY	n1, n2, ⓓ			7-395
时间检查	TIMCHK	n1, n2, ⓓ	• 对输入条件的 ON 时间进行计测，如果连续 ON 的时间超出了所设置的时间，则将ⓓ中指定的软元件置为 ON。		7-397
1 字节单位的直接读取 / 写入	ZRRDB	n, ⓓ			7-399
	ZRWRB	n, ⓢ			7-402
	ADRSET	ⓢ, ⓓ			7-405
通过键盘的数字键输入	KEY	ⓢ, n, ⓓ, ⓟ	• 将 ASCII 数据输入到ⓢ中指定的输入模块 8 点中，转换为 16 进制数值后存储到ⓓ中指定的软元件编号以后。		7-406
变址寄存器的批量保存	ZPUSH	ⓓ	• 将变址寄存器 Z0 ～ Z15 的内容保存到ⓓ中指定的软元件以后。		7-410
变址寄存器的批量恢复	ZPOP	ⓓ	• 将ⓓ中指定的软元件以后中保存的数据，读取到变址寄存器 Z0 ～ Z15 中。		
模块信息读取	UNIRD	n1, n2, ⓓ	• 从 (n1) 中指定的起始输入输出编号开始，将 (n2) 中指定的点数的模块信息，存储到ⓓ中指定的软元件以后。		7-413
模块型号读取	TYPERD	n, ⓓ	• 对 n 中指定的起始输入输出编号的模块型号进行读取后，存储到ⓓ中指定的软元件以后。		7-419

表 2.39 其它指令（续）

分类	指令名	自变量	处理内容	执行条件	说明页面
跟踪设置	TRACE	－	• 将外围设备中设置的跟踪数据，在 SM800、SM801、SM802 为 ON 时按照所设置的次数，存储到采样跟踪用文件中。		7-424
跟踪复位	TRACER	－	• 对通过 TRACE 指令设置的数据进行复位。		
至指定文件的数据写入	SP_FWRITE	⑤0，⑤1，②，⑩0，⑪	• 对指定的文件进行数据写入。		7-427
从指定文件中进行数据读取	SP_FREAD	⑤0，⑤1，⑩0，⑪，⑫	• 从指定的文件中进行数据读取。		7-438
至标准 ROM 的数据写入	SP_DEVST	⑤1，②，n，ⓓ	• 对标准 ROM 的软元件数据存储用文件进行数据写入。		7-451
从标准 ROM 中进行数据读取	S_DEVLD	Ⓢ，n，ⓓ	• 从标准 ROM 的软元件数据存储用文件中进行数据读取。		7-454
从存储器进行程序装载	PLOADP	Ⓢ，ⓓ	• 将存储卡、标准存储器（除驱动器 0 以外）中存储的程序传送至驱动器 0 中后，置为待机状态。		7-456
从程序存储器中进行程序卸载	PUNLOADP	Ⓢ，ⓓ	• 将程序存储器（驱动器 0）中存储的待机程序从存储器中删除。		7-460
装载 + 卸载	PSWAPP	⑤1，②，ⓓ	• 将⑤1中指定的程序存储器（驱动器 0）中存储的待机程序从存储器中删除后，将②中指定的存储卡、标准存储器（除驱动器 0 以外）中存储的程序传送至驱动器 0 中后，置为待机状态。		7-463
文件寄存器高速块传送	RBMOV	Ⓢ，n，ⓓ	• 将Ⓢ中指定的软元件算起 n 点的 16 位数据，批量传送至ⓓ中指定的软元件开始的 n 点中。		7-466
用户信息	UMSG	Ⓢ	• 将指定的字符串，作为显示模块中的用户信息进行显示。		7-471

2.6 数据链接用指令

2.6.1 网络刷新指令

表 2.40 网络刷新指令

分类	指令名	自变量	处理内容	执行条件	说明页面
网络刷新	S_ZCOM_J	Jn*	• 对指定网络进行刷新处理。		8-2
	S_ZCOM_U	Un*			

2.6.2 路由信息的读取 / 登录指令

表 2.41 路由信息的读取 / 登录指令

分类	指令名	自变量	处理内容	执行条件	说明页面
路由信息的读取	S_RTREAD	n, ⓓ	• 对路由参数中设置的数据进行读取。		8-7
路由信息的登录	S_RTWRITE	n, Ⓢ	• 对路由参数中指定的区域进行路由数据的写入。		8-9

2.7 多 CPU 间专用指令

2.7.1 至本站 CPU 共享存储器的写入指令

表 2.42 至本站 CPU 共享存储器的写入指令

分类	指令名	自变量	处理内容	执行条件	说明页面
本站 CPU 共享存储器写入	S_TO	ⓢ1, ⓢ2, ⓢ3, ⓢ4, ⓓ	• 将本站的软元件写入到本站 CPU 模块的 CPU 共享存储器中。		9-2
	TO	Ⓢ, n1, n2, n3	• 将本站的软元件写入到本站 CPU 模块的 CPU 共享存储器中。		
	DTO	Ⓢ, n1, n2, n3	• 将本站的软元件以 32 位为单位写入到本站 CPU 模块的 CPU 共享存储器中。		

2.7.2 从其它机号 CPU 共享存储器中的读取指令

表 2.43 从其它机号 CPU 共享存储器中的读取指令

分类	指令名	自变量	处理内容	执行条件	说明页面
其它机号 CPU 共享存储器读取	FROM	n1, n2, n3, ⓓ	• 从其它机号 CPU 模块的 CPU 共享存储器中将软元件读取到本站中。		9-13
	DFRO	Ⓢ, n1, n2, n3	• 从其它机号 CPU 模块的 CPU 共享存储器中将软元件以 32 位为单位读取到本站中。		

2.8 多 CPU 间高速通信专用指令

2.8.1 多 CPU 间高速通信专用指令

表 2.44 多 CPU 间高速通信专用指令

分类	指令名	自变量	处理内容	执行条件	说明页面
至其它机号的软元件写入	D_DDWR	n1,Ⓢ1, Ⓢ2, Ⓓ1, Ⓓ2	• 多 CPU 系统配置时，将本站 CPU 中指定的软元件 Ⓢ2 以后的数据，以Ⓢ1+1 中指定的写入数据点数，存储到其它机号 CPU(n1) 的指定的软元件以后。		10-11
	DP_DDWR	n1,Ⓢ1, Ⓢ2, Ⓓ1, Ⓓ2			
从其它机号的软元件读取	D_DDRD	n1,Ⓢ1, Ⓢ2, Ⓓ1, Ⓓ2	• 多 CPU 系统配置时，将其它机号 CPU(n1) 的指定的软元件以后的数据，以Ⓢ1+1 中指定的读取数据点数，存储到本站 CPU 中指定的软元件 Ⓢ2 以后。		10-15
	DP_DDRD	n1,Ⓢ1, Ⓢ2, Ⓓ1, Ⓓ2			

3 指令的构成

3.1 指令的构成

对于 CPU 模块中可使用的指令，可以分为指令名及自变量。

指令名及自变量的用途如下所示。

• 指令名 ..... 表示该指令的功能。
• 自变量 ..... 表示指令中使用的输入输出数据。

自变量有源数据、目标数据、软元件数、执行条件、执行状态。

(1) 源ⓢ

(a) 源是运算中使用的数据。

(b) 根据指令中指定的软元件，其情况如下所示。

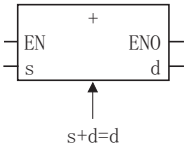
• 常数 ..... 对运算中使用的数值进行指定。
是在创建程序时进行设置，在程序的执行过程中不能进行变更。
将常数用于可变数据中的情况下，应进行变址修饰。

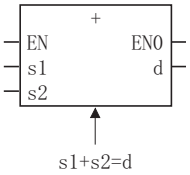
• 位软元件、字软元件 ..... 对存储运算中使用的数据的软元件进行指定。
在执行运算之前需要预先将数据存储到指定的软元件中。
在程序执行过程中，通过对指定的软元件中存储的数据进行变更，可以对该指令中使用的数据进行变更。

(c) 对于使用位软元件的源，不能直接输入触点。

(2) 目标ⓓ

(a) 目标中用于存储运算后的数据。
但是，根据指令有时需要在运算前在目标中存储运算中使用的数据。
例 : BIN16 位数据的加法运算指令的情况下





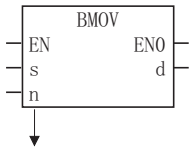
(b) 对目标必须设置用于存储数据的软元件。

(c) 在存储位软元件的目标中，不能直接连接线圈。

(3) 软元件数 / 传送数 (n)

(a) 在软元件数 / 传送数中，对使用多个软元件的指令中使用的软元件数 / 传送数进行指定。

例 : 块传送指令的情况下



n: 对通过BMOV指令传送的传送数进行指定。

(b) 软元件数 / 传送数的允许设置范围为 0 ～ 32767。

但是，软元件数 / 传送数为 0 的情况下，该指令将无处理。

(4) 执行条件 (EN)

输入变量 EN 输入指令的执行条件。

(5) 执行状态 (ENO)

输出变量 ENO 输出执行状态。

要 点

关于标签及结构体等的指令构成的详细内容，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

3.2 编程时的注意事项

在 CPU 模块中执行基本指令、应用指令时，在下述情况下将变为运算出错状态。

- 存在各指令的说明页面中记载的出错的情况下。
- 使用智能功能模块软元件时，指定的输入输出编号位置处未安装智能功能模块的情况下。
- 使用智能功能模块软元件时，指定的缓冲存储器地址不存在的情况下。
- 使用链接软元件时，相应网络不存在的情况下。
- 使用链接软元件时，指定的输入输出编号位置处未安装网络模块的情况下。
- 使用多 CPU 间共享软元件时，指定 CPU 模块的起始输入输出编号位置处未安装 CPU 模块的情况下。（仅通用型 QCPU(Q00UJCPU 除外)）
- 使用多 CPU 间共享软元件时，指定的共享存储器地址不存在的情况下。（仅通用型 QCPU(Q00UJCPU 除外)）
- 进行了跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的设置的情况下。（使用通用型 QCPU(Q00UJCPU 除外)、LCPU 时）

要 点

未安装进行文件寄存器设置的存储卡时，或未进行文件寄存器设置时，如果进行了至文件寄存器的读取 / 写入，将变为以下情况。

(1) 高性能型 QCPU 的情况下

即使进行了至文件寄存器的读取 / 写入也不会变为出错状态。但是，如果从文件寄存器进行读取，将存储 “0H”。

(2) 通用型 QCPU、LCPU 的情况下

如果对文件寄存器进行读取 / 写入，将变为 OPERATION ERROR（出错代码：4101）。

(1) 软元件范围检查

在 CPU 模块中基本指令、应用指令中使用的软元件的范围检查情况如下所示。

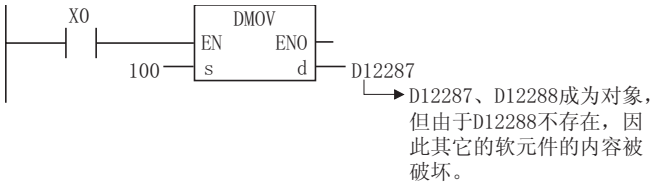
(a) 处理固定长度的软元件的指令 (MOV、DMOV 等)

1) 基本型 QCPU、高性能型 QCPU 的情况下

不进行软元件范围的检查。

超出了相应软元件范围的情况下，数据将被写入到其它软元件中。*1

例如，数据寄存器被分配为 12k 点的情况下，即使超出了 D12287 也不变为出错状态。



进行了变址修饰的情况下也不进行软元件范围检查。

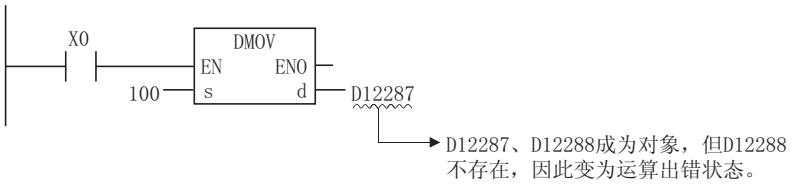
进行变址修饰的结果超出了相应软元件范围的情况下，数据将被写入到其它软元件中。*1

*1 ： 关于内部用户软元件的分配顺序，请参阅本节 (c) 字符串数据。

2) 通用型 QCPU、LCPU 的情况

进行软元件范围的检查。超出了相应软元件范围的情况下，将变为运算出错状态。

例如，数据寄存器被分配为 12k 点的情况下，超出了 D12287 时将变为出错状态。



进行了变址修饰的情况下将进行软元件范围检查。

此外，根据可编程控制器参数也可以设置为不进行软元件范围检查。*2

*2: 关于变址修饰时不进行软元件范围检查的设置，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

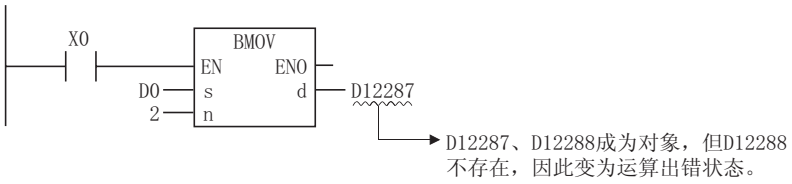
(b) 处理可变长度的软元件的指令（指定传送数的 BMOV、FMOV 等）

1) 基本型 QCPU、高性能型 QCPU 的情况

进行软元件范围的检查。

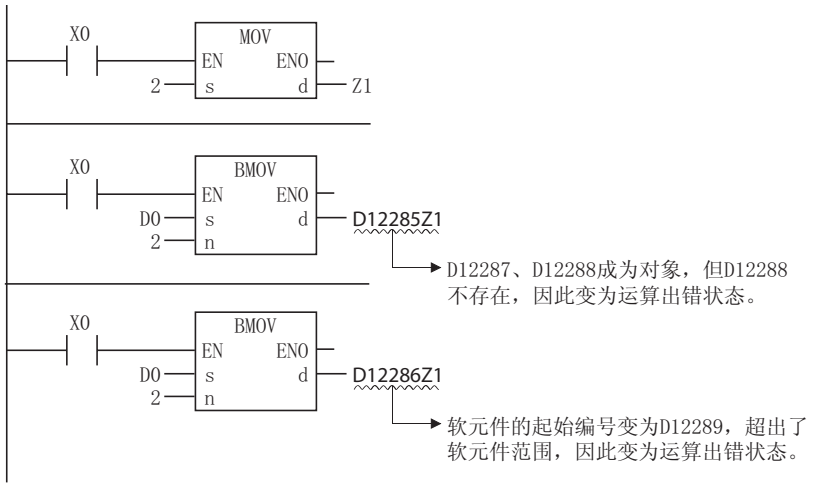
超出了相应软元件范围的情况下，将变为运算出错状态。

例如，数据寄存器被分配为 12k 点的情况下，超出了 D12287 时将变为出错状态。



进行了变址修饰的情况下将进行软元件范围检查。

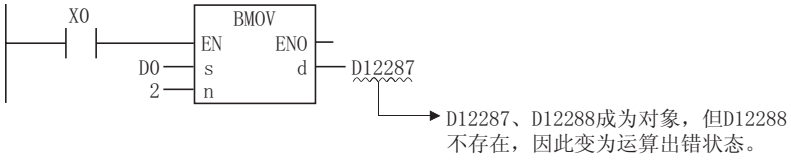
但是，进行了变址修饰的情况下，软元件的起始编号超出了相应软元件范围时将变为出错状态。



2）通用型 QCPU、LCPU 的情况

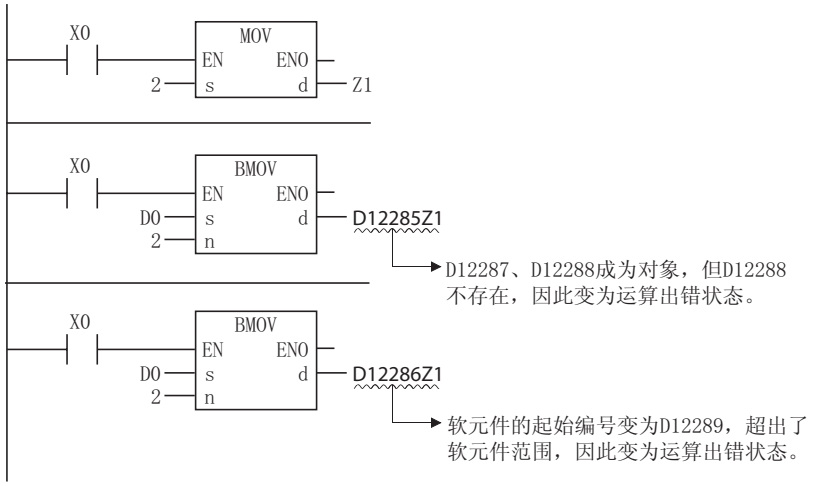
进行软元件范围的检查。超出了相应软元件范围的情况下，将变为运算出错状态。

例如，数据寄存器被分配为 12k 点的情况下，超出了 D12287 时将变为出错状态。



进行了变址修饰的情况下将进行软元件范围检查。

但是，进行了变址修饰的情况下，软元件的起始编号超出了相应软元件范围时将变为出错状态。



此外，通过可编程控制器参数也可以设置为不进行软元件范围检查。*2

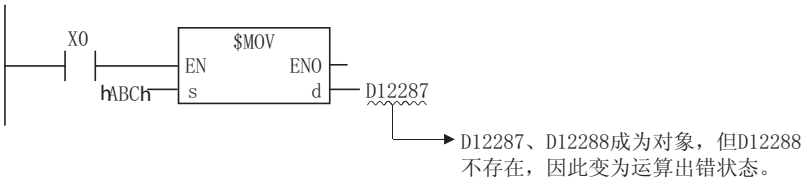
*2: 关于变址修饰时不进行软元件范围检查的设置，请参阅所使用的 CPU 模块的用户手册（功能解说／程序基础篇）。

(c) 字符串数据

由于字符串数据均为可变长度处理，因此进行软元件的范围检查。

超出了相应软元件范围的情况下，将变为运算出错状态。

例如，数据寄存器被分配为 12k 点的情况下，超出了 D12287 时将变为出错状态。



但是，在基本型 QCPU、高性能型 QCPU 中进行变址修饰，软元件的起始编号超出了软元件范围的情况下，不变为运算出错状态而对其它的软元件进行访问。
在通用型 QCPU、LCPU 中进行了下述访问的情况下，将变为出错状态。（出错代码：4101）

1) 由于变址修饰，超出了软元件的边界的访问。
（区域 A 的范围）
各软元件的分配顺序如下所示。

2) 由于修饰，超出了文件寄存器的边界的访问。

3) 未设置文件寄存器文件时，至文件寄存器 (R、ZR) 的访问。

4) 超出了文件寄存器文件的范围的至文件寄存器 (R、ZR) 的访问。

此外，通过在可编程控制器参数中设置为不进行变址修饰时的软元件范围检查，可以设置为即使进行了 1) ～ 4) 的访问，也不进行出错检测。
但是，根据通用型 QCPU 的序列号，其动作有如下表所示的不同。*2

变址修饰时的软元件范围设置	通用型 QCPU 的序列号的前 5 位数		LCPU
	序列号 “10021” 以前	序列号 “10022” 以后	
实施	1) ～ 4) 的访问时检测出出错		
不实施	2) ～ 4) 的访问时检测出出错	不检测出错	

*2: 关于不进行变址修饰时的软元件范围检查的设置，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

要 点

在通用型 QCPU、LCPU 中，不能指定跨越内部用户软元件 (SW) 与文件寄存器 (R) 之间的变址修饰。(出错代码： 4101。)

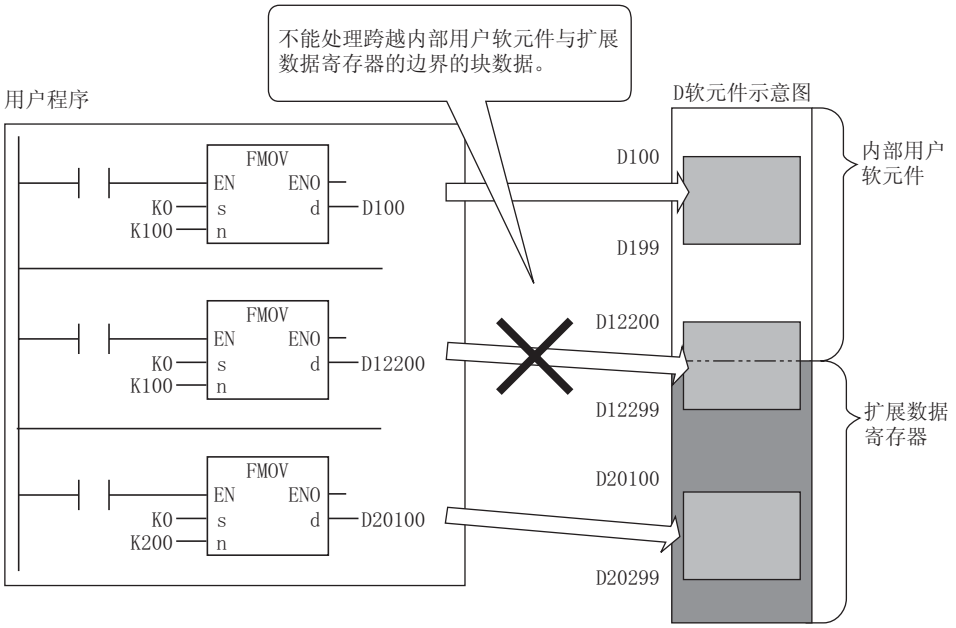
备 注

关于内部用户软元件的分配变更请参阅所使用的 CPU 模块的用户手册 (功能解说 / 程序基础篇)。

(d) 在直接访问输出 (DY) 中进行了变址修饰的情况下，进行软元件的范围检查。

(e) 使用扩展数据寄存器 (D)、扩展链接寄存器 (W) 时的注意事项
(除 Q00UJCPU 以外的通用型 QCPU、LCPU)
在如下所示的指定方法中，不能进行跨越内部用户软元件及扩展数据寄存器 (D)、扩展链接寄存器 (W) 的边界的指定。如果指定将变为 OPERATION ERROR(出错代码： 4101)。

- 变址修饰指定
- 间接指定
- 处理块数据的指令中的指定 *1



*1: 块数据是指如下所示的数据：
- 通过 FMOV、BMOV、BK+ 等将多个字作为运算对象的指令所处理的数据。
- 通过 SP.FWRITE、SP.FREAD 等指定的 2 字以上构成的控制数据。
- 32 位以上的数据格式的数据 (BIN32 位、实数、软元件的间接地址)

(2) 软元件的数据检查

在 CPU 模块中基本指令、应用指令中使用的软元件的数据检查情况如下所示。

(a) BIN 数据的情况
即使运算结果上溢、下溢也不变为出错状态。
此时进位标志也不变为 ON。

(b) BCD 数据的情况
1) 对各个位是否为 BCD 值 (0 ～ 9) 进行检查。
各个位超出了 0 ～ 9 的范围 (A ～ F) 的情况下将变为运算出错状态。
2) 即使运算结果上溢、下溢也不变为出错状态。
此时进位标志也不变为 ON。

(c) 浮动小数点数据的情况
1) 在进行单精度浮动小数点运算的指令中，运算结果为下述情况下将变为运算出错状态。
1.0×2^{-127} 以下时
1.0×2^{128} 忙以下时
2) 在进行双精度浮动小数点运算的指令中，运算结果为下述情况下将变为运算出错状态。
1.0×2^{-1023} 以下时
1.0×2^{1024} 以下时

(d) 字符串数据的情况
不进行数据的检查。

(3) 至缓冲存储器的访问
对于至缓冲存储器的访问，建议通过使用了智能功能模块软元件 (Un\G0 ～) 的指令进行访问。

(4) 至多 CPU 间共享存储器的访问
对于至多 CPU 间共享存储器的访问，建议通过使用了多 CPU 间共享软元件 (U3En\G10000 ～) 的指令进行访问。

3.3 指令的执行条件

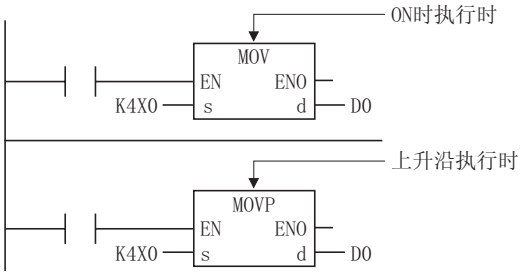
CPU 模块的顺控程序指令、基本指令、应用指令的执行条件中有如下所示的 4 种。

- 常时执行 .... 指令的执行与软元件的 ON/OFF 无关的指令。
  例： LD=
- ON 时执行.... 在输入条件为 ON 的状态下执行的指令。
  例： MOV 指令、FROM 指令
- 上升沿执行 .. 仅在输入条件的上升沿 (OFF → ON) 时执行的指令。
  PLS 指令、MOVP 指令
- 下降沿执行 .. 仅在输入条件的下降沿 (ON → OFF) 时执行的指令。
  PLF 指令

在相当于线圈的基本指令、应用指令中，在同一指令中有 “ON 时执行”及 “上升沿执行”的 2 种可能的情况下，在指令名的后面附加 “P”对执行条件加以区别。

- ON 时执行的指令 ..... 指令名
- 上升沿时执行指令.... 指令名 P

在 MOV(P) 指令中，ON 时执行与上升沿执行按下述方式指定。



3.4 使用同一软元件的 OUT 指令、SET/RST 指令、PLS/PLF 指令时的动作

以下对使用了同一软元件的 OUT 指令、SET/RST 指令、PLS/PLF 指令在 1 个扫描中执行了次时的动作进行说明。

(1) 同一软元件的 OUT 指令的情况

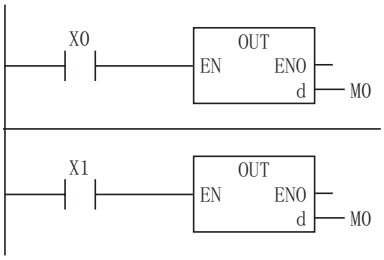
在 1 个扫描中不要多次执行同一软元件的 OUT 指令。

在 1 个扫描中对同一软元件的 OUT 指令执行了多次的情况下，执行各个 OUT 指令时，根据 OUT 指令之前的运算结果，指定软元件执行 ON/OFF。

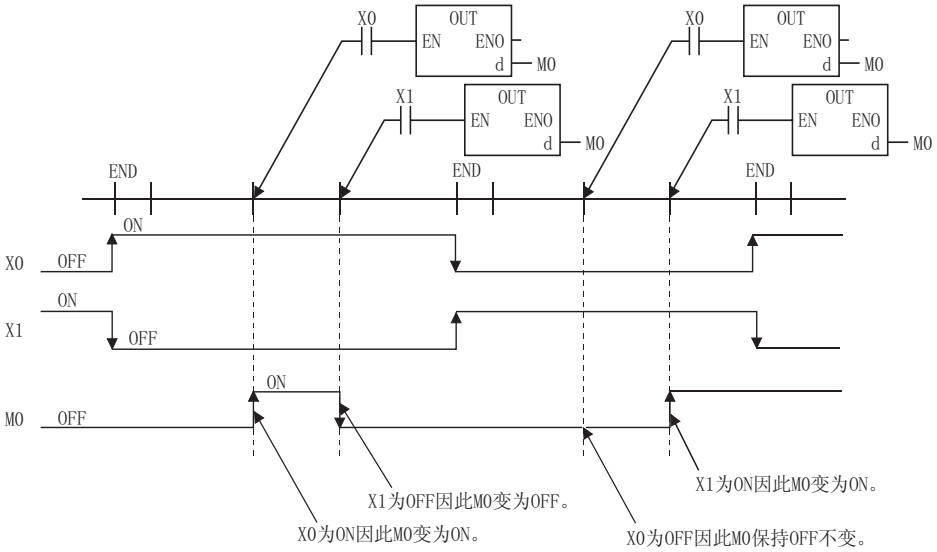
执行各个 OUT 指令时，决定指定软元件的 ON/OFF，因此 1 个扫描中有可能反复执行 ON/OFF。

在输入的 X0 及 X1 中，创建了使同一内部继电器 (M0)ON/OFF 的梯形图时的动作如下图所示。

[梯形图]



[时序图]



刷新类型的 CPU 模块的情况下，如果在 OUT 指令中指定输出 (Y)，则 1 个扫描的最后执行的 OUT 指令的 ON/OFF 状态将被输出。

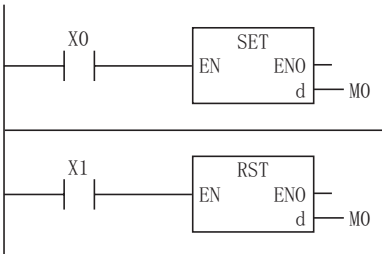
(2) 使用了同一软元件的 SET/RST 指令的情况

(a) SET 指令在执行指令为 ON 时将指定软元件置为 ON，在执行指令为 OFF 时执行无处理。
因此，在 1 个扫描中执行了多次同一软元件的 SET 指令的情况下，只要 1 个执行指令为 ON，则指定软元件将变为 ON。

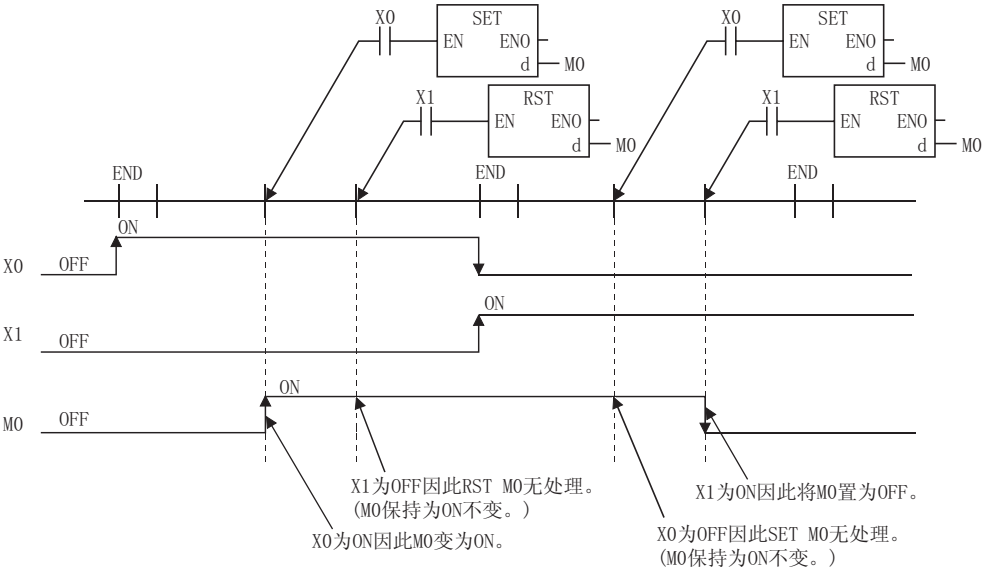
(b) RST 指令在执行指令为 ON 时将指定软元件置为 OFF，在执行指令为 OFF 时执行无处理。
因此，在 1 个扫描中执行了多次同一软元件的 RST 指令的情况下，只要 1 个执行指令为 ON，则指定软元件将变为 OFF。

(c) 在 1 个扫描存在有同一软元件的 SET 指令及 RST 指令的情况下，SET 指令在执行指令为 ON 时将指定软元件置为 ON，RST 指令在执行指令为 ON 时将指定软元件置为 OFF。
SET 指令及 RST 指令的执行指令为 OFF 的情况下，指定软元件的 ON/OFF 状态不变化。

[梯形图]



[时序图]



刷新类型的 CPU 模块的情况下，如果在 SET/RST 指令中指定输出 (Y)，则 1 个扫描的最后执行的 SET/RST 指令的 ON/OFF 状态将被输出。

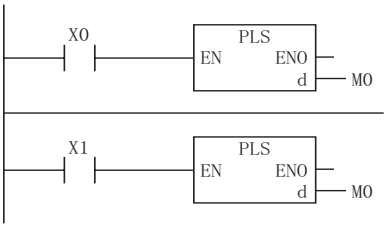
(3) 使用了同一软元件的 PLS 指令的情况

PLS 指令在执行指令的 OFF → ON 时将指定软元件置为 ON。除 OFF → ON 以外（OFF → OFF、ON → ON、ON → OFF）时，将指定软元件置为 OFF。

将同一软元件的 PLS 指令在 1 个扫描中执行了多次的情况下，各 PLS 指令的执行指令 OFF → ON 时，将指定软元件置为 ON。除 OFF → ON 以外时将指定软元件置为 OFF。

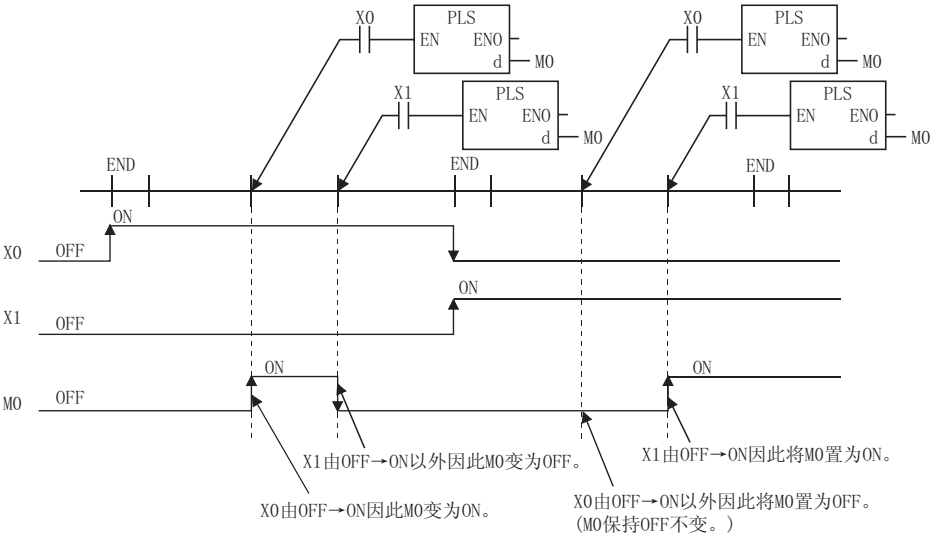
因此将同一软元件的 PLS 指令在 1 个扫描中执行了多次的情况下，通过 PLS 指令变为 ON 的软元件有可能 1 个扫描不变为 ON。

[梯形图]

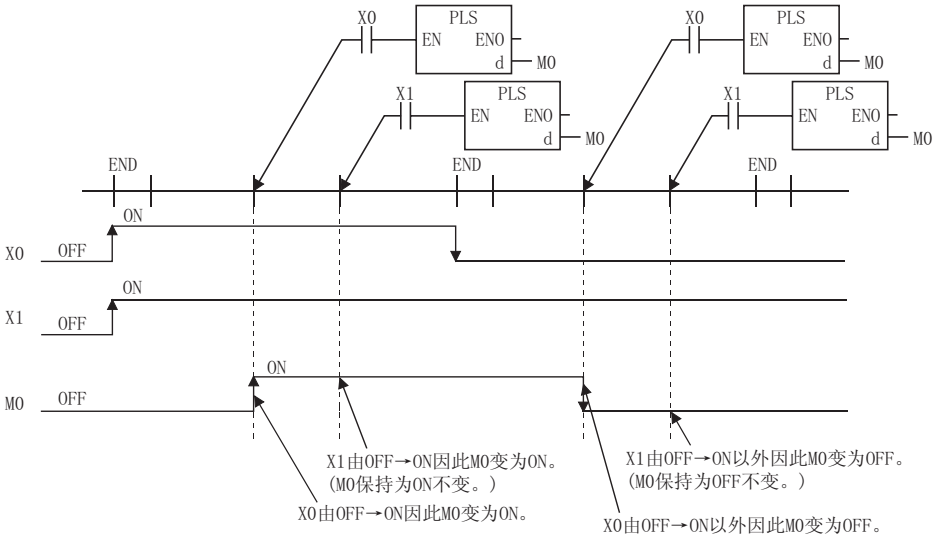


[时序图]

• X0 与 X1 的 ON/OFF 时机不相同时（指定软元件 1 个扫描不变为 ON。）



• X0 与 X1 的 OFF → ON 时机相同时



刷新类型的 CPU 模块的情况下，如果通过 PLS 指令指定输出 (Y)，则 1 个扫描的最后执行的 PLS 指令的 ON/OFF 状态将被输出。

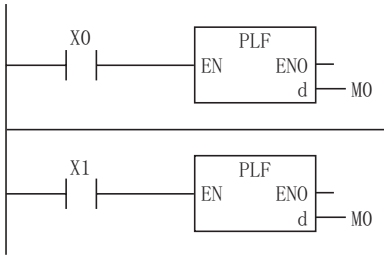
(4) 使用了同一软元件的 PLF 指令的情况

PLF 指令在执行指令的 ON → OFF 时将指定软元件置为 ON。在除 ON → OFF 以外 (OFF → OFF、OFF → ON、ON → ON) 时，将指定软元件置为 OFF。

将同一软元件的 PLF 指令在 1 个扫描中执行了多次的情况下，各 PLF 指令的执行指令由 ON → OFF 时，将指定软元件置为 ON。除 ON → OFF 以外时将指定软元件置为 OFF。

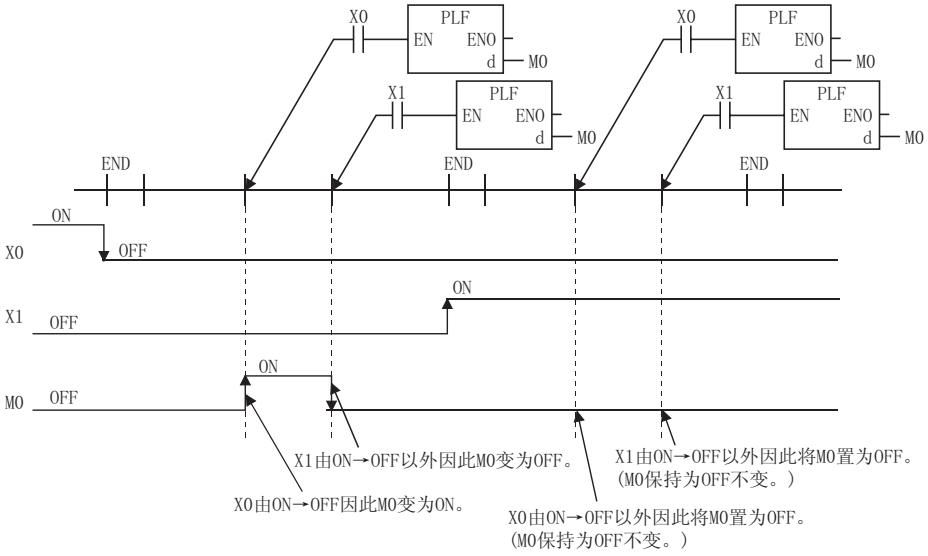
因此，将同一软元件的 PLF 指令在 1 个扫描中执行了多次的情况下，通过 PLF 指令变为 ON 的软元件有可能 1 个扫描不变为 ON。

[梯形图]

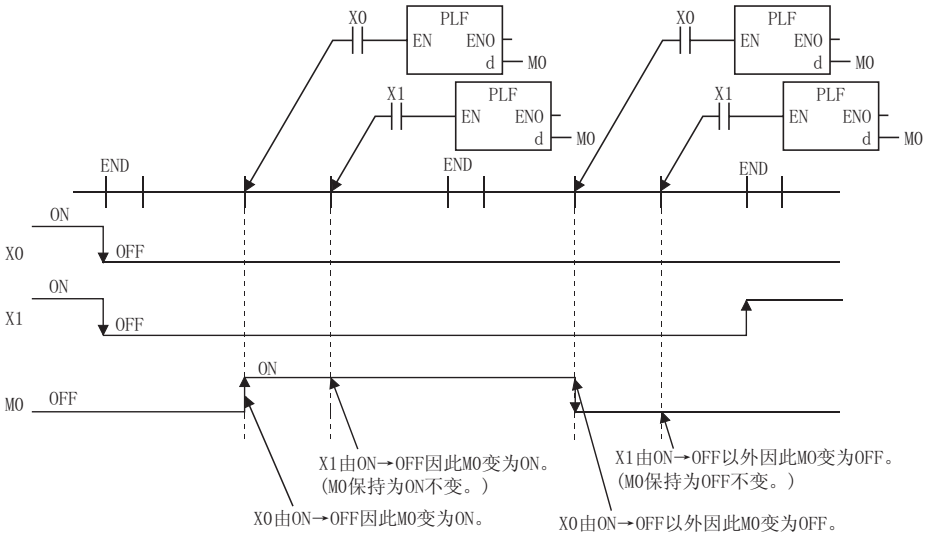


[时序图]

• X0 与 X1 的 ON/OFF 时机不相同时（指定软元件 1 个扫描不变为 ON。）



• X0 与 X1 的 ON → OFF 时机相同时



刷新类型的 CPU 模块的情况下，通过 PLF 指令指定输出 (Y) 时，1 个扫描的最后执行的 PLF 指令的 ON/OFF 状态将被输出。

3.5 使用文件寄存器时的注意事项

以下介绍在 QCPU(Q 模式)、LCPU 中使用文件寄存器时的注意事项。

(1) 不能使用文件寄存器的 CPU 模块

Q00JCPU、Q00UJCPU 不能使用文件寄存器。
使用文件寄存器时，应使用除 Q00JCPU、Q00UJCPU 以外的 CPU 模块。

(2) 使用文件寄存器的设置

使用文件寄存器时，需要对可编程控制器参数或 QDRSET 指令中使用的文件寄存器进行设置。
(Q00CPU、Q01CPU、LCPU 的可编程控制器参数被设置为 “使用文件寄存器”，因此无需设置。此外，在 LCPU 中，不能使用 QDRSET 指令。)
未对使用的文件寄存器进行设置的情况下，在使用了文件寄存器的指令中将无法正常运算。

要 点

即使未在可编程控制器参数中对使用的文件寄存器进行设置，也可创建使用了文件寄存器的程序。此外，在除通用型 QCPU、LCPU 以外的 CPU 模块中，即使将该程序写入到 CPU 模块中并执行也不会变为出错状态。
但是，不能对文件寄存器进行正确的数据读取 / 写入，因此应加以注意。
在通用型 QCPU、LCPU 中，如果执行使用了文件寄存器的程序，将变为出错状态。

(3) 文件寄存器区域的预留

(a) 高性能型 QCPU、通用型 QCPU(Q00UJCPU 除外)
使用文件寄存器时，将文件寄存器登录到标准 RAM / 存储卡中，对文件寄存器的区域进行预留。

(b) 基本型 QCPU(Q00JCPU 除外)
文件寄存器的区域在标准 RAM 中被事先预留。
用户无需对文件寄存器区域进行预留。

(c) LCPU
使用文件寄存器时，将文件寄存器登录到标准 RAM 中，对文件寄存器的区域进行预留。

在各 CPU 模块中可使用文件寄存器的存储器如下表所示。

存储器	高性能型 QCPU、通用型 QCPU (Q00UJ 除外)	基本型 QCPU(Q00J 除外)、LCPU
标准 RAM	○	○
存储卡 *1	○ *2	×

○：可以登录； ×：不能登录

*1: 使用快闪存储器时，只能从文件寄存器中进行读取。(不能对快闪 ROM 进行写入。)
*2: 在 Q00UCPU、Q01UCPU 中不能使用存储卡。

备 注

关于文件寄存器的设置方法、文件寄存器区域的预留方法，请参阅所使用的 CPU 模块的用户手册（功能解説 / 程序基础篇）。

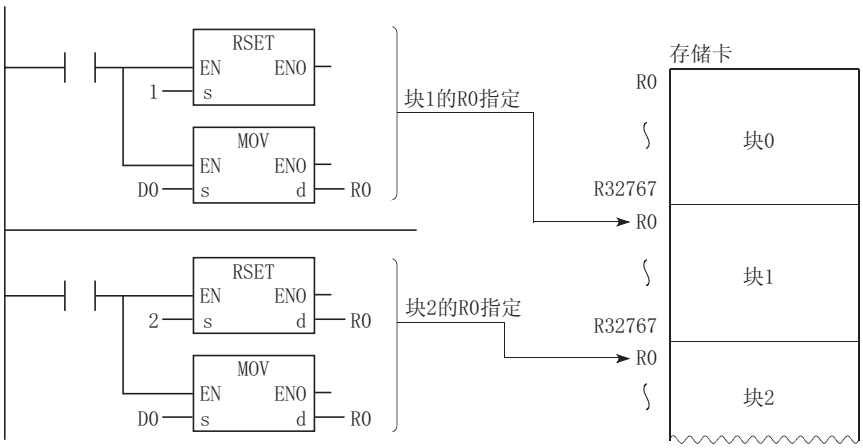
(4) 文件寄存器编号的指定超出了所登录的点数时

(a) 基本型 QCPU、高性能型 QCPU 的情况
即使对超出所登录的点数的文件寄存器编号进行读取 / 写入，也不会变为出错状态。
但是，将不能对文件寄存器进行正确的数据读取 / 写入，因此应加以注意。

(b) 通用型 QCPU、LCPU 的情况
对超出所登录的点数的文件寄存器编号进行读取 / 写入时，将变为出错状态。(出错代码 : 4101)

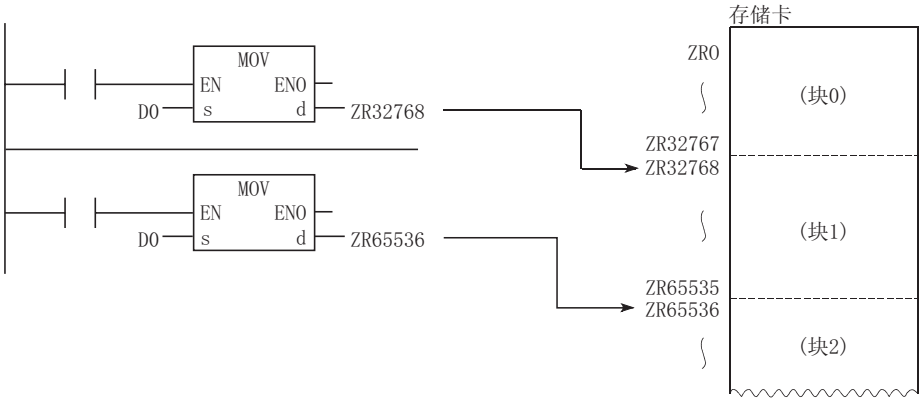
(5) 文件寄存器的指定方法
文件寄存器的指定方法中，有块切换方式及连号访问方式。

(a) 块切换方式
块切换方式是指，将所使用的文件寄存器点数以 32k 点 (1 块) 为单位进行分割指定的方式。
对于 32k 点以上的文件寄存器，通过 RSET 指令对使用的文件寄存器的块号进行切换指定。
各块均在 R0 ～ R32767 范围内进行指定。



(b) 连号访问方式

连号访问方式是指，将超出 32k 点的文件寄存器以连续的软元件编号进行指定的方式。可以将多个块的文件寄存器作为连续的文件寄存器使用。



(6) 将文件寄存器指定为刷新软元件时的设置及限制

(a) 刷新软元件的设置

在下述设置中，可以进行刷新软元件的设置。

- CC-Link IE 控制网络模块的刷新设置（在 LCPU 中不能设置。）
- MELSECNET/H 的刷新设置（在 LCPU 中不能设置。）
- CC-Link 的刷新设置
- 智能功能模块的自动刷新设置
- 多 CPU 系统的自动刷新设置（在 LCPU 中不能设置。）

(b) 限制事项

将文件寄存器指定为刷新软元件的情况下，应注意以下 1) ～ 3) 的限制事项。

1) 在 QCPU(Q 模式）的可编程控制器参数中，指定时使用了与程序名相同名称的文件寄存器的情况下，将无法进行正确的刷新。

使用与程序名相同名称的文件寄存器的情况下，与程序设置中被设置为最后编号的程序同名的文件寄存器将被刷新。希望对刷新数据进行读写的情况下，应使用 QDRSET 指令，切换为相应的文件寄存器之后进行指定。

2) 通过 QDRSET 指令对文件寄存器的文件名进行了变更的情况下，以及对驱动器 No. 进行了变更的情况下，将无法正确地进行刷新。（在 LCPU 中不能使用 QDRSET 指令。）

通过 QDRSET 指令对文件名及驱动器 No. 进行了变更的情况下，在执行 END 时设置文件将被链接刷新。希望对刷新数据进行读写的情况下，应指定执行 END 时的设置文件。

但是，在除通用型 QCPU 以外的 CPU 中，软元件被指定为 “ZR”的情况下，如果通过 QDRSET 指令对驱动器 No. 进行变更，将变为 LINK PARA ERROR(3101) 状态，应加以注意。（软元件被指定为 “R”的情况下不会变为出错状态。）

3) 通过 RSET 指令对块号进行了切换的情况下，切换后的块号的文件寄存器 (R) 将被刷新。

通过 RSET 指令对块号进行了切换的情况下，执行 END 时的块号的文件寄存器 (R) 将被刷新。希望对刷新数据进行读写的情况下，应指定执行 END 时的块号。

(7) 使用快闪存储器中的文件寄存器时的注意事项

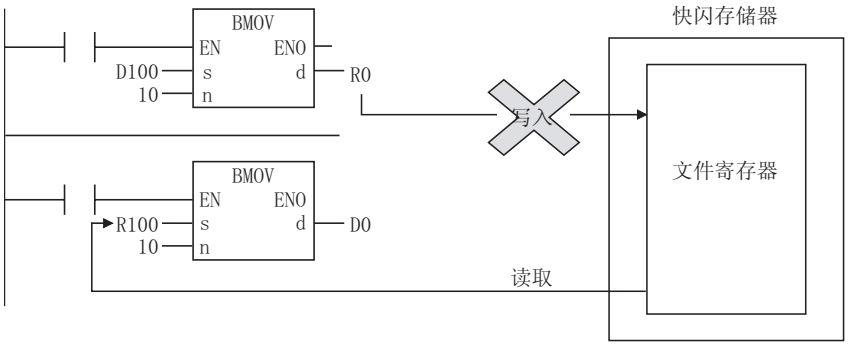
以下介绍可使用文件寄存器的快闪存储器的注意事项有关内容。

(a) 快闪存储器如下所示。

• Flash 卡

(b) 对于快闪存储器中的文件寄存器只能通过顺控程序进行读取。

（不能通过顺控程序对快闪存储器进行写入。）

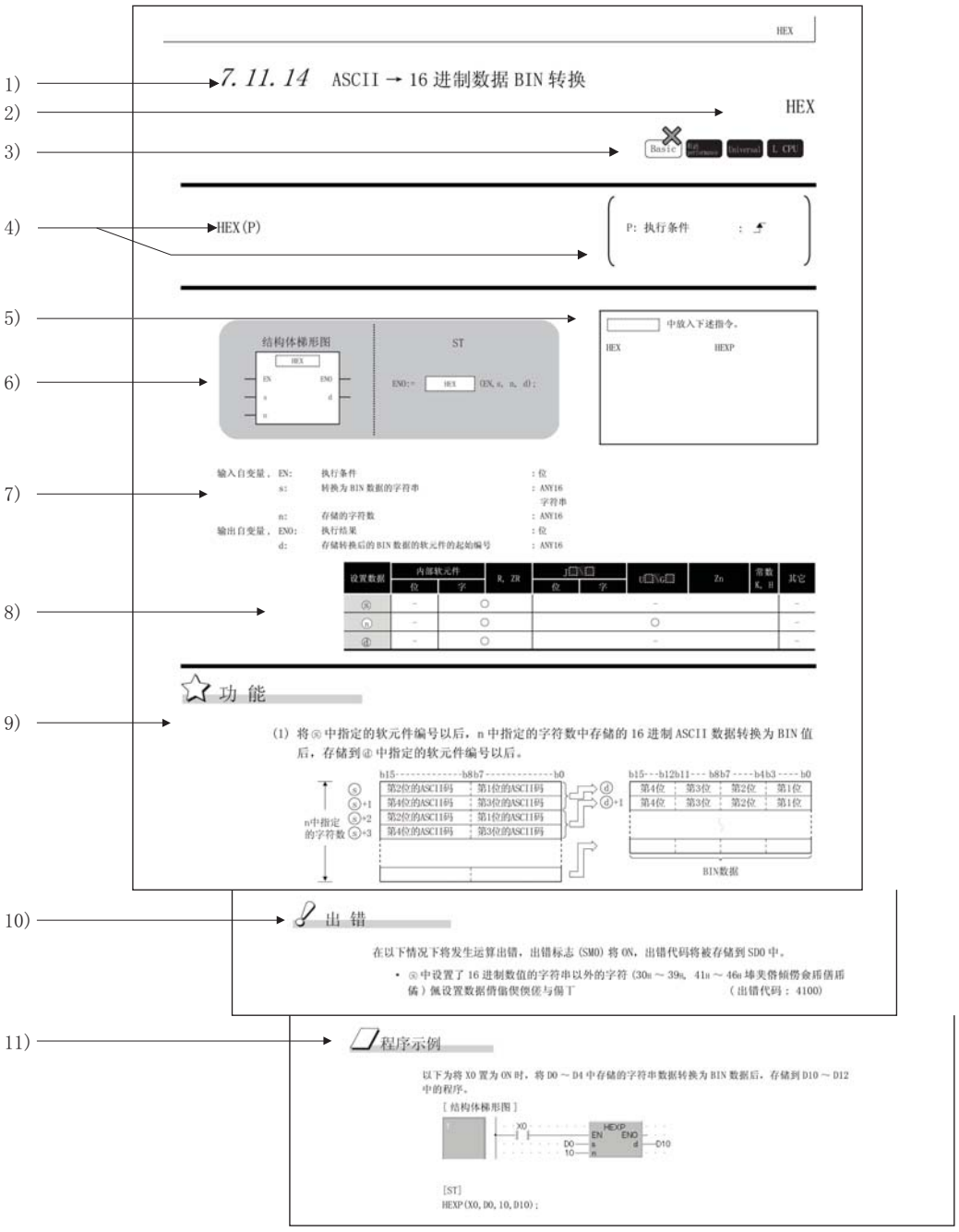


将快闪存储器作为文件寄存器使用的情况下，应预先进行数据写入。

(c) 通过编程工具对 Flash 卡进行数据写入。

4 指令的阅读方法

从本手册的第 5 章以后，将以如下所示的构成对各指令进行说明。



1) 表示项编号、指令的概要。

2) 表示说明对象的指令。

3) 表示可使用指令的 CPU 模块。

图标				内容
基本型 QCPU	高性能型 QCPU	通用型 QCPU	LCPU	
				对于通常的图标，表示可以使用相应的指令。
				对于带 Ver. 符号的图标，表示可在有部分限制的情况下使用。（功能版本、软元件版本）
				对于带 × 符号的图标，表示不能使用相应的指令。

4) 表示指令名及指令的执行条件。

执行条件	常时执行	ON 中执行	ON 时执行 1 次	OFF 中执行	OFF 时执行 1 次
说明页面的记载符号	无记入				

5) 表示可记述的指令名。

6) 表示结构体梯形图、ST 语言中的指令的记述格式。

7) 表示指令的输入自变量、输出自变量的名称、各自变量的数据类型以及软元件的直接指定可否。

关于各数据类型的详细内容，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

8) 指令中可使用的软元件附带有○符号。

可使用的软元件的用途分类如下所示。

软元件分类	内部软元件（系统、用户）		文件寄存器 R,ZR	链接直接软元件 *4*6 J□\□		智能功能模块 U□\G□	变址寄存器 Zn	常数 *5	其它 *5
	位	字		位	字				
可使用的软元件 *1	X,Y,M,L,SM,F,B,SB,FX,FY *2	T,ST,C, *3 D,W,SD,SW,FD	R,ZR	J□\X J□\Y J□\B J□\SB	J□\W J□\SW	U□\G□	Z	K,H,E,$	P,I,J,U,DX,DY,N,BL,TR,BL\S,V

*1 : 关于各软元件的说明，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

*2: FX、FY 只能使用位数据，FD 只能使用字数据。

*3: 将 T、ST、C 用于除下述指令以外时，只能作为字数据使用。
（不能作为位数据使用。）
[可作为位数据使用的指令]
LD,LDI,AND,ANI,OR,ORI,LDP,LDF,ANDP,ANDF,ORP,ORF,LDPI,LDFI,ANDPI,ANDFI,ORPI,ORFI,OUT,RST,BKRST

*4: 在 CC-Link IE 控制网络、MELSECNET/H,MELSECNET/10 中可以使用。

*5: 在常数、其它栏中，记载可设置的软元件。

*6: 链接直接软元件 (J□\□) 不能在 LCPU 中使用。

9) 表示指令担当的功能。

10) 表示出错的有无。有出错的情况下，表示引起出错的条件有关内容。

11) 表示结构体梯形图、ST 的程序示例。

备忘录

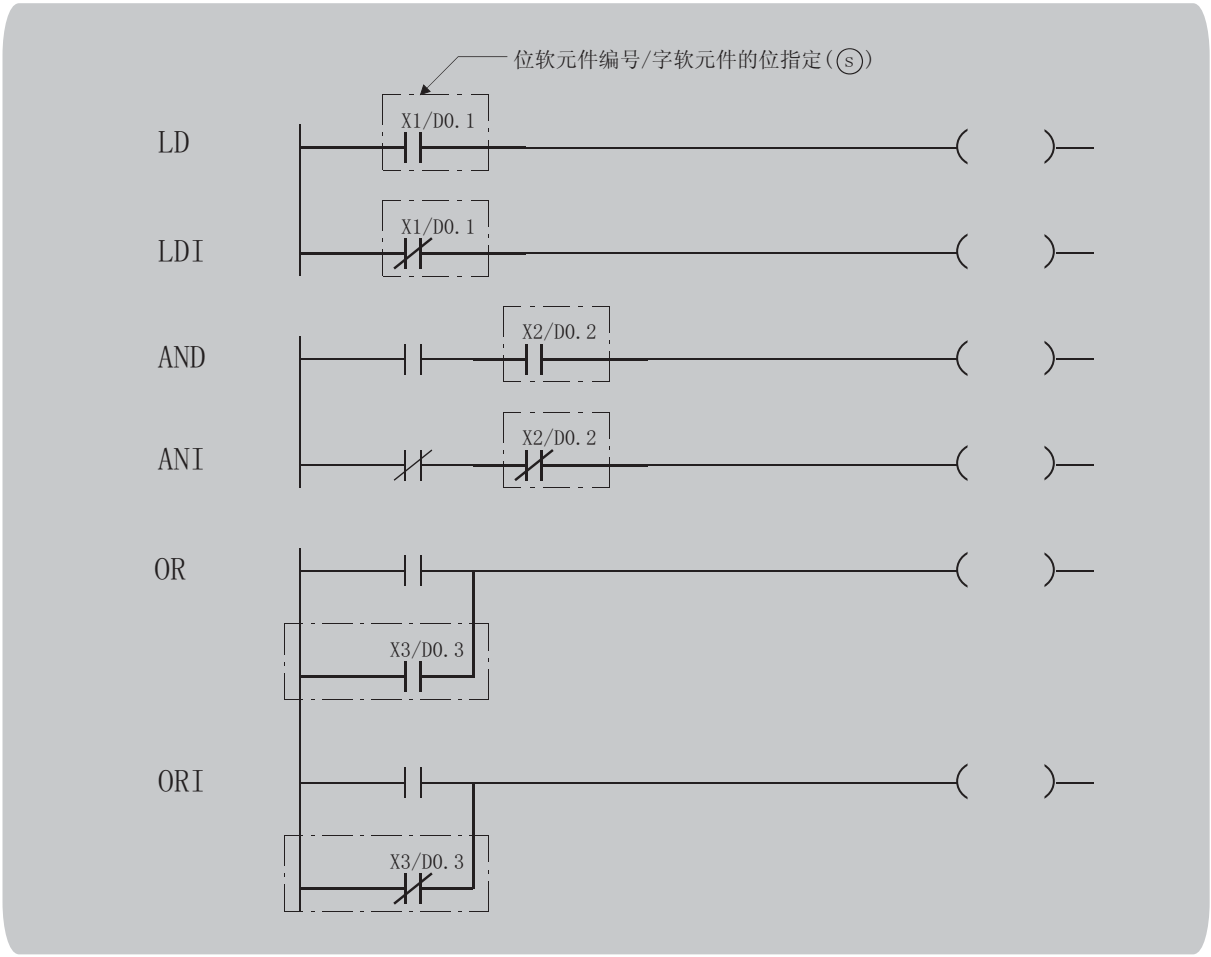
5 顺控程序指令

5.1 触点指令

5.1.1 运算开始、串联连接、并联连接

LD, LDI, AND, ANI, OR, ORI

Basic | High performance | Universal | L CPU



ⓢ : 作为触点使用的软元件(位)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DX, BL
	位	字		位	字				
ⓢ	○						-		○

功 能

LD, LDI

LD 是常开触点运算开始指令、LDI 是常闭触点运算开始指令，用于对指定软元件的 ON/OFF 信息 *1 进行获取，作为运算结果。

*1 : 字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

AND,ANI

AND 是常开触点串联连接指令、ANI 是常闭触点串联连接指令，用于对指定位软元件的 ON/OFF 信息 *2 进行获取，与到此为止的运算结果进行 AND 运算，将该值作为运算结果。

*2:　字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

OR,ORI

OR 是 1 个常开触点的并联连接指令、ORI 是 1 个常闭触点的并联连接指令，用于对指定软元件的 ON/OFF 信息 *3 进行获取，与到此为止的运算结果进行 OR 运算，将该值作为运算结果。

*3 :　字软元件的位指定时，根据指定位的 1/0 而 ON/OFF。

备 注

字软元件的位指定时，位的指定是以 16 进制数进行。
对于 D0 的 b11，将变为 D0.0B。
关于字软元件的位指定，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

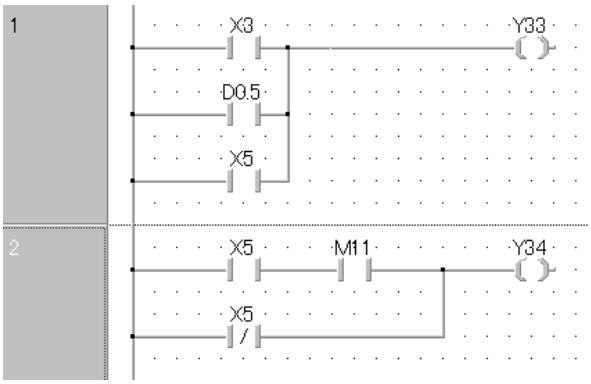
出 错

不存在 LD、LDI、AND、ANI、OR、ORI 指令相关的运算出错。

程序示例

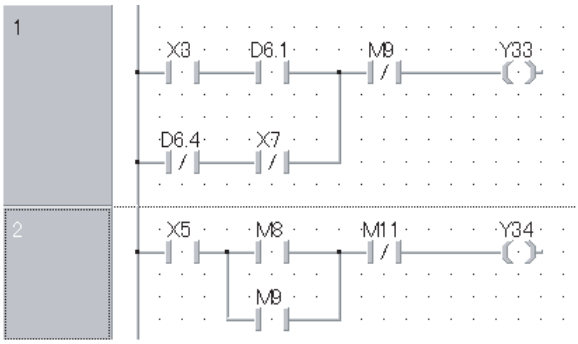
(1) 使用了 LD、AND、OR、ORI 指令的程序。

[结构体梯形图]



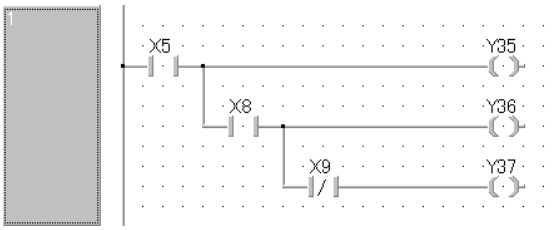
(2) 通过 ANB、ORB 指令进行触点合并的程序。

[结构体梯形图]



(3) OUT 指令的并联程序。

[结构体梯形图]



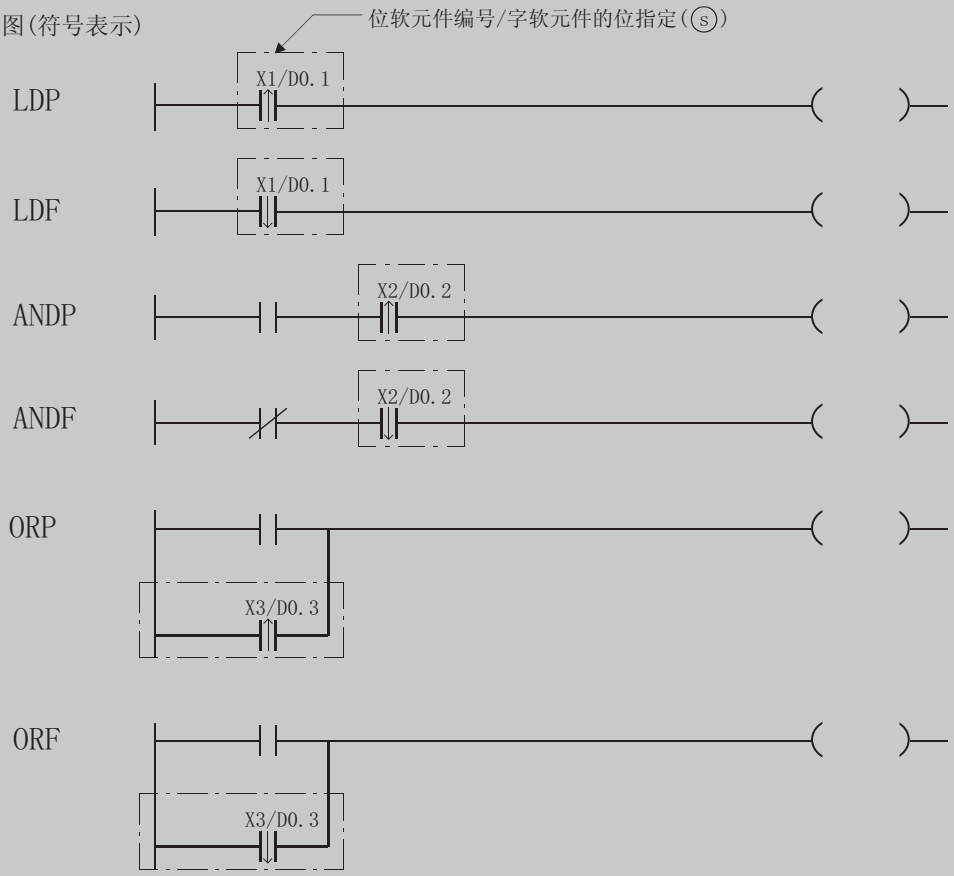
5.1.2 脉冲运算开始、脉冲串联连接、脉冲并联连接

LDP, LDF, ANDP, ANDF, ORP, ORF

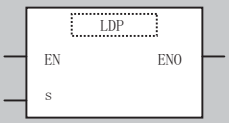
Basic | High performance | Universal | L CPU

LD(P)(F)
AND(P)(F)
OR(P)(F)

结构体梯形图(符号表示)



结构体梯形图
(功能表示)



ST

ENO:= LDP (EN, s);

□ 中放入下述指令。

LDP	LDF
ANDP	ANDF
ORP	ORF

输入自变量， EN: 执行条件 :位
s: 作为触点使用的软元件 :位
输出自变量， ENO: 执行结果 :位

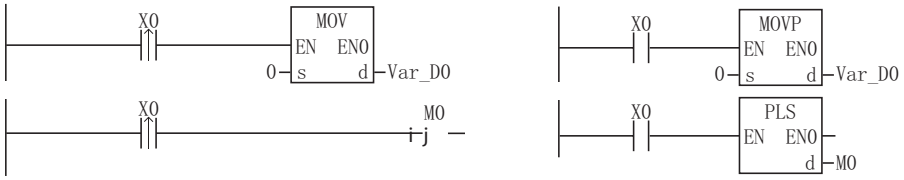
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DX
	位	字		位	字				
ⓢ			○				-		○

功 能

LDP,LDF

(1) LDP 是上升沿脉冲运算开始指令，仅在指定位软元件的上升沿时 (OFF → ON) 导通。字软元件的位指定时，仅在指定位发生 0 → 1 的变化时导通。

只有 LDP 指令的情况下，与 ON 中执行指令的脉冲化指令 (□P) 相同。



(2) LDF 是下降沿脉冲运算开始指令，在指定位软元件的下降沿时 (ON → OFF) 导通。

字软元件的位指定时，指定位发生了 1 → 0 的变化时导通。

ANDP,ANDF

(1) ANDP 是上升沿脉冲串联连接指令，ANDF 是下降沿脉冲串联连接指令，用于与到此为止的运算结果进行 AND 运算，作为运算结果。

ANDP、ANDF 中使用的 ON/OFF 信息如下表所示。

ANDP、ANDF 中指定的软元件		ANDP 状态	ANDF 状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	ON	OFF
OFF	0	OFF	
ON	1		
ON → OFF	1 → 0		ON

ORP,ORF

(1) ORP 是上升沿脉冲并联连接指令，ORF 是下降沿脉冲串联连接指令，用于与到此为止的运算结果进行 OR 运算后作为运算结果。

ORP、ORF 中使用的 ON/OFF 信息如下表所示。

ORP、ORF 中指定的软元件		ORP 状态	ORF 状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	ON	OFF
OFF	0	OFF	
ON	1		
ON → OFF	1 → 0		ON

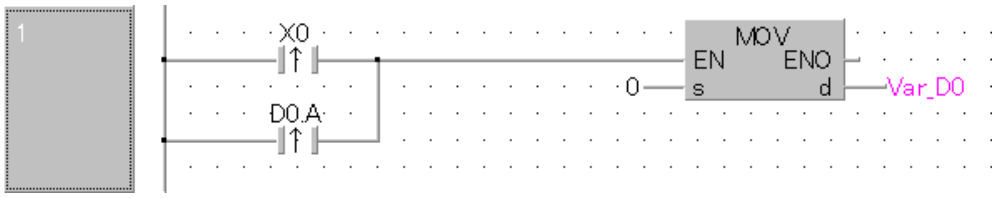
出 错

不存在 LDP、LDF、ANDP、ANDF、ORP、ORF 指令相关的运算出错。

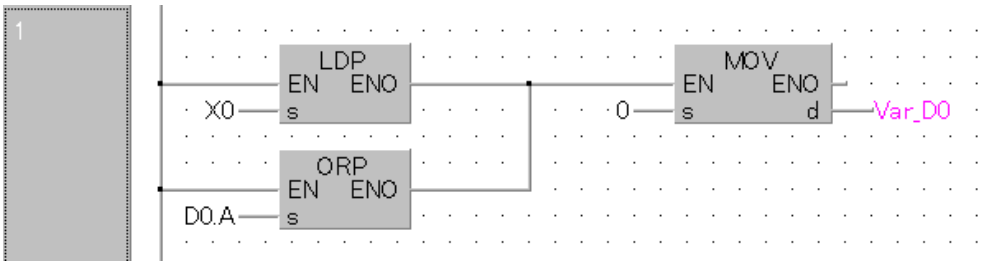
程序示例

该程序是输入 X0 或数据寄存器的 D0 的 b10(位 11) 上升沿时，执行 MOV 指令的程序。

[结构体梯形图 (符号表示)]



[结构体梯形图 (功能表示)]




```
[ST]
IF (LDP(TRUE,X0) OR LDP(TRUE,D0.A)) THEN
    MOV(TRUE,0,Var_D0);
END_IF;
```

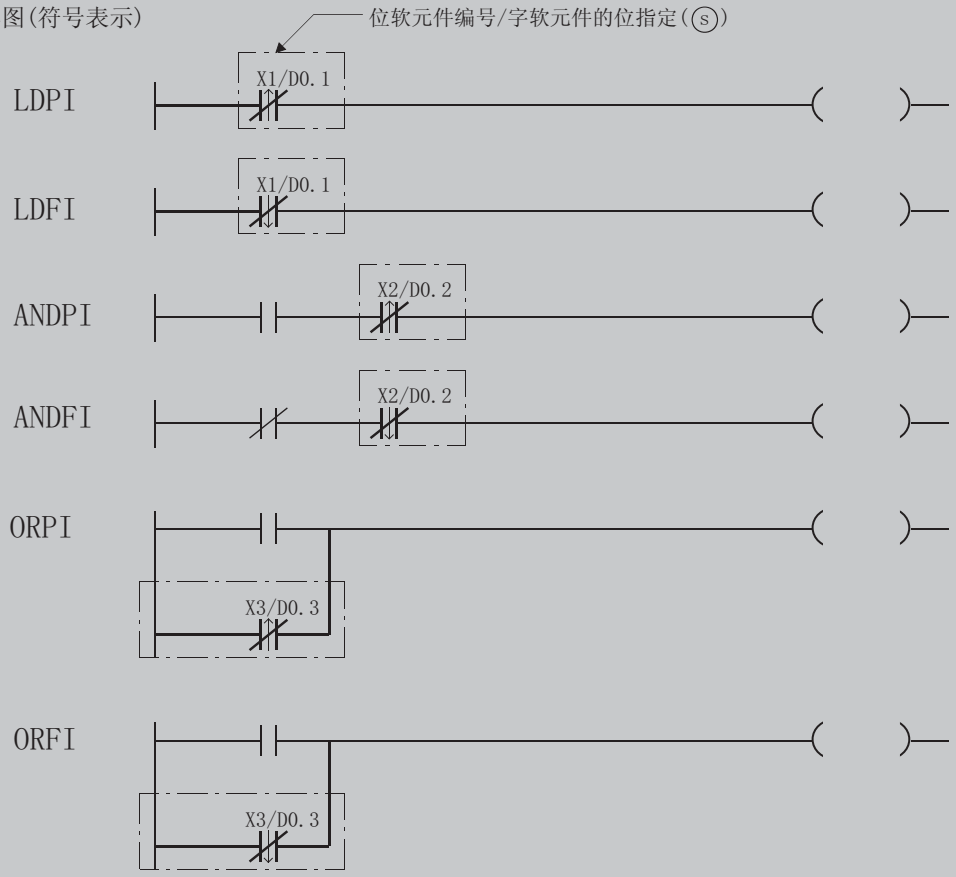
5.1.3 脉冲否运算开始、脉冲否串联连接、脉冲否并联连接

LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI

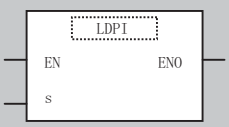
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

LD(P)(F)I
AND(P)(F)I
OR(P)(F)I

结构体梯形图(符号表示)



结构体梯形图
(功能表示)



ST

```
ENO:= LDPI (EN, s);
```

□ 中放入下述指令。

LDPI	LDFI
ANDPI	ANDFI
ORPI	ORFI

输入自变量， EN:　　执行条件　　：位
　　s:　　作为触点使用的软元件　　：位
输出自变量， ENO:　　执行结果　　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DX
	位	字		位	字				
ⓢ	○	--	○	○	--	○	--		○

功 能

LDPI,LDFI

(1) LDPI 是上升沿脉冲否运算开始指令，在指定位软元件的 OFF 时，ON 时，下降沿时 (ON → OFF) 导通。字软元件的位指定时，指定位为 0 的情况下、为 1 的情况下、发生 1 → 0 的变化的情况下导通。

(2) LDFI 是下降沿脉冲否运算开始指令，在指定位软元件的上升沿时 (OFF → ON)、OFF 时、ON 时导通。字软元件的位指定时，指定位为 0 的情况下、为 1 的情况下，发生 0 → 1 的变化的情况下导通。

LDPI、LDFI 中指定的软元件		LDPI 状态	LDFI 状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

ANDPI,ANDFI

(1) ANDPI 是上升沿脉冲否串联连接指令，ANDFI 是下降沿脉冲否串联连接指令，用于与到此为止的运算结果进行 AND 运算，作为运算结果。
ANDPI、ANDFI 中使用的 ON/OFF 信息如下表所示。

ANDPI、ANDFI 中指定的软元件		ANDPI 状态	ANDFI 状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

ORPI,ORFI

(1) ORPI 是上升沿脉冲否并联连接指令，ORFI 是下降沿脉冲否并联连接指令，用于与到此为止的运算结果进行 OR 运算，作为运算结果。
ORPI、ORFI 中使用的 ON/OFF 信息如下表所示。

ORPI、ORFI 中指定的软元件		ORPI 状态	ORFI 状态
位软元件	字软元件的位指定		
OFF → ON	0 → 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON → OFF	1 → 0	ON	OFF

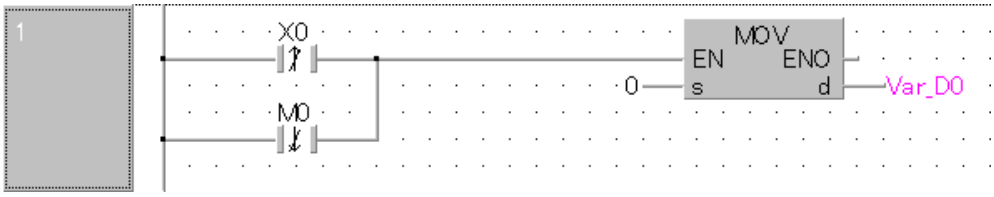
出 错

(1) 不存在 LDPI、LDFI、ANDPI、ANDFI、ORPI、ORFI 指令相关的运算出错。

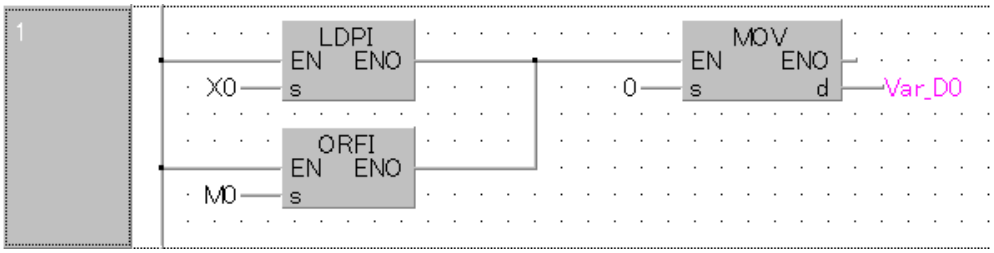
程序示例

(1) 以下为 X0 为 ON/OFF/ON → OFF 时，或 M0 为 ON/OFF/OFF → ON 时，在 Var_D0 中存储 0 的程序。

[结构体梯形图（符号表示）]



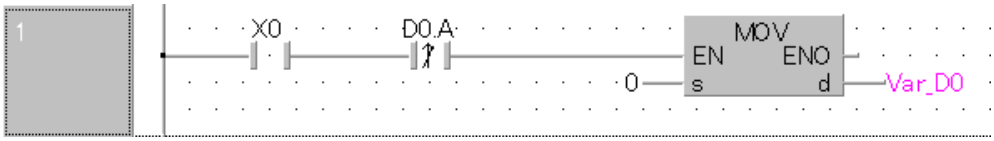
[结构体梯形图（功能表示）]



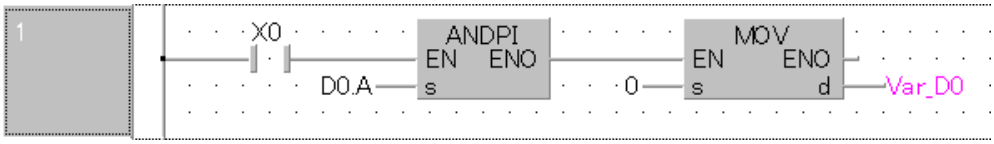

```
[ST]
IF (LDPI(TRUE,X0) OR LDFI(TRUE,M0)) THEN
     MOV(TRUE,0,Var_D0);
END_IF;
```

(2) 以下为 X0 为 ON，且 D0 的 b10（位 11）为 ON/OFF/ON → OFF 时，在 Var_D0 中存储 0 的程序。

[结构体梯形图（符号表示）]



[结构体梯形图（功能表示）]



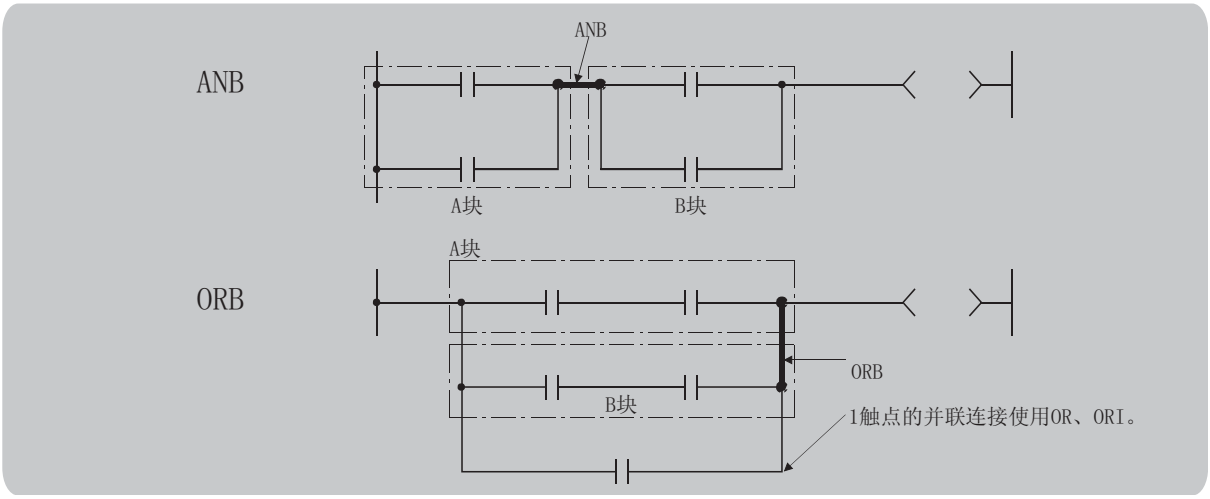

```
[ST]
IF (X0 & LDPI(TRUE,D0.A)) THEN
     MOV(TRUE,0,Var_D0);
END_IF;
```

5.2 合并指令

5.2.1 梯形图块串联连接、并联连接

ANB, ORB

Basic | High performance | Universal | L CPU



设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

ANB

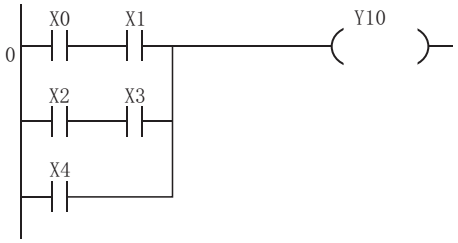
(1) 用于进行 A 块与 B 块的 AND 运算，作为运算结果。

(2) ANB 符号不是触点符号，而是连接符号。

ORB

(1) 用于进行 A 块与 B 块的 OR 运算，作为运算结果。

(2) ORB 进行 2 触点以上的梯形图块的并联连接。
对于仅 1 个触点的并联连接，使用 OR、ORI，无需使用 ORB。
[结构体梯形图]



(3) ORB 符号不是触点符号而是连接符号。

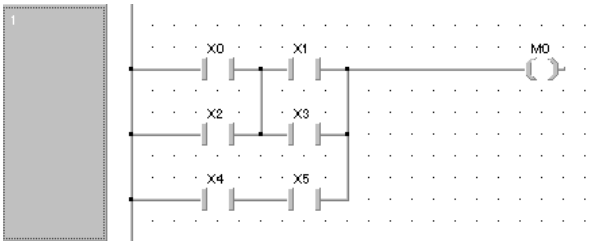
出 错

不存在 ANB、ORB 指令相关的运算出错。

程序示例

以下为使用了 ANB、ORB 的程序。

[结构体梯形图]

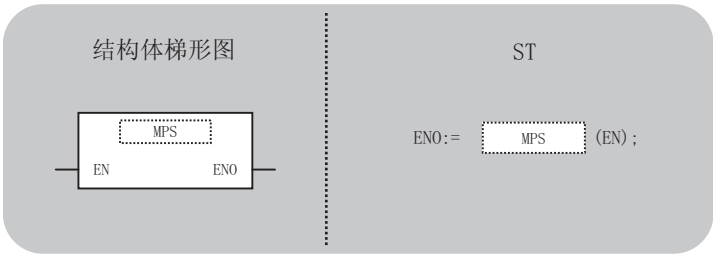


5.2.2 运算结果推进、读取、弹出

MPS, MRD, MPP

Basic | High performance | Universal | L CPU

MPS
MRD
MPP



□ 中放入下述指令。

MPS
MRD
MPP

输入自变量，EN:　　执行条件　　：位
输出自变量，ENO:　　执行结果　　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

MPS

(1) 用于存储 MPS 指令之前的运算结果 (ON/OFF)。

(2) MPS 指令最多可以连续使用 16 次。
在中途使用了 MPP 指令的情况下对 MPS 指令的使用数进行 -1。

MRD

用于对通过 MPS 指令存储的运算结果进行读取，从下一步开始以该运算结果执行运算。

MPP

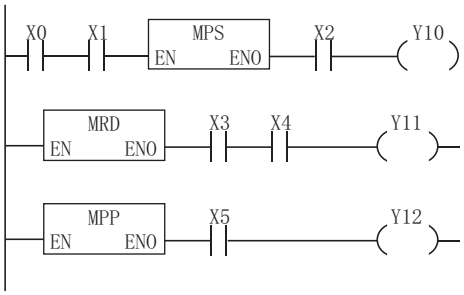
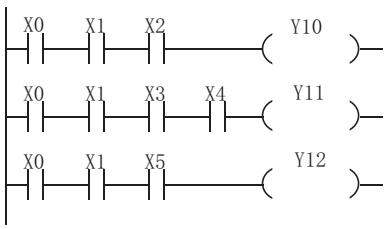
(1) 用于对通过 MPS 指令存储的运算结果进行读取，从下一步开始以该运算结果执行运算。

(2) 对通过 MPS 指令存储的运算结果进行清除。

(3) 对 MPS 指令的使用数进行 -1。

要 点

1. 使用了 MPS、MRD、MPP 时的情况及未使用时的情况的梯形图如下所示。

使用了 MPS、MRD、MPP 时的梯形图	未使用 MPS、MRD、MPP 时的梯形图
	

2. 应将 MPS、MPP 指令的使用数设置为相同。
   MPS、MPP 指令的使用数不相同的情况下，编程工具的梯形图模式中将无法正常地显示梯形图。

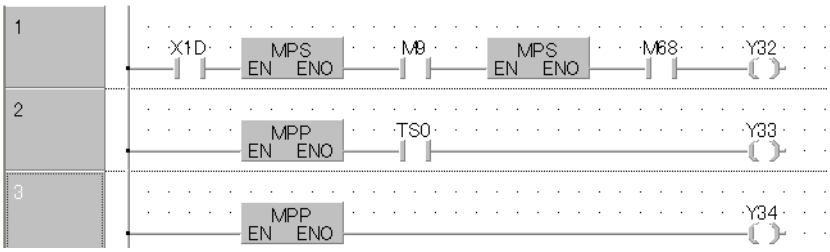
出 错

不存在 MPS、MRD、MPP 指令相关的出错。

程序示例

(1) 使用了 MPS、MRD、MPP 的程序。

[结构体梯形图]

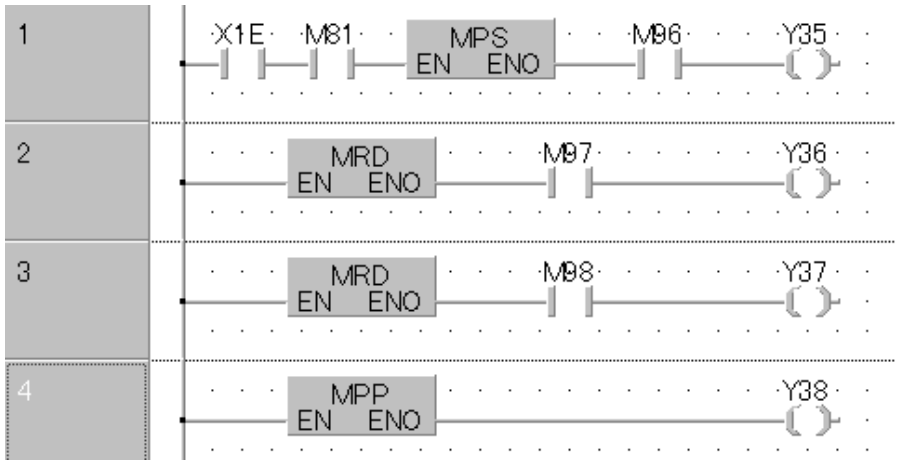


[ST]

```
OUT(MPS(MPS(X1D) AND M9) AND M68,Y32);
OUT(MPP(TRUE) AND T50,Y33);
OUT(MPP(TRUE),Y34);
```

(2) MPS、MPP 连续的程序。

[结构体梯形图]



[ST]

```
OUT(MPS(X1E AND M81) AND M96,Y35);
OUT(MRD(TRUE) AND M97,Y36);
OUT(MRD(TRUE) AND M98,Y37);
OUT(MPP(TRUE),Y38);
```

5.2.3 运算结果取反

INV

Basic | High performance | Universal | L CPU

INV

结构体梯形图

ST

ENO:= INV (EN);

中放入下述指令。

INV

输入自变量， EN: 执行条件 :位

输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

对 INV 指令之前的运算结果进行取反。

INV 指令之前的运算结果	执行 INV 指令后的运算结果
OFF	ON
ON	OFF

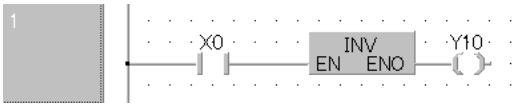
出 错

不存在 INV 指令相关的运算出错。

程序示例

以下为对 X0 的 ON/OFF 数据进行取反后，通过 Y10 输出的程序。

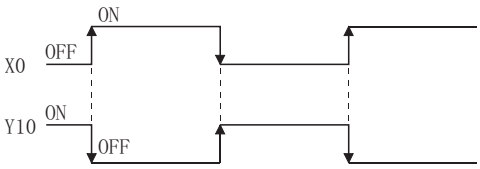
[结构体梯形图]



[ST]

```
OUT(INV(X0),Y10);
```

[时序图]



要 点

INV 指令是以 INV 指令之前的运算结果执行动作，因此应与 AND 指令在同一位置处使用。

INV 指令不能在 LD、OR 的位置处使用。

5.2.4 运算结果脉冲化

MEP, MEF

Basic | High performance | Universal | L CPU

MEP
MEF

结构体梯形图

ST

ENO:= MEP (EN);

[ ] 中放入下述指令。

MEP
MEF

输入自变量， EN: 执行条件 :位
输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

MEP

(1) 在 MEP 指令之前的运算结果的上升沿时 (OFF → ON)，变为 ON(导通状态)。
在 MEP 指令之前的运算结果为上升沿以外的情况下，变为 OFF(非导通状态)。

(2) 如果使用 MEP 指令，在多个触点串联连接的情况下脉冲化处理将变得容易。

MEF

(1) 在 MEF 指令之前的运算结果的下降沿时 (ON → OFF)，变为 ON(导通状态)。
在 MEF 指令之前的运算结果为下降沿以外的情况下，变为 OFF(非导通状态)。

(2) 如果使用 MEF 指令，在多个触点串联连接的情况下脉冲化处理将变得容易。

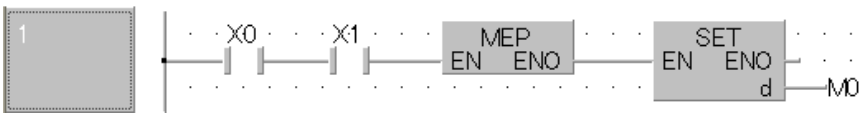
出 错

不存在 MEP、MEF 指令相关的运算出错。

程序示例

以下为对 X0 与 X1 的运算结果进行脉冲化的程序。

[结构体梯形图]




```
[ST]
IF X0 AND X1 THEN
    MEP(TRUE);
    SET(TRUE,M0);
END_IF;
```

要 点

1. 对于 MEP、MEF 指令，如果通过子程序或 FOR ～ NEXT 指令等对进行了变址修饰的触点执行脉冲化，将有可能无法正常动作。
   对通过子程序、FOR ～ NEXT 指令进行了变址修饰的触点执行脉冲化时，应使用 EGP、EGF 指令。
2. 对于 MEP、MEF 指令，由于是以 MEP、MEF 指令之前的运算结果执行动作，因此应在与 AND 指令相同的位置处使用。
   对于 MEP、MEF 指令，不能在 LD、OR 的位置处使用。

5.2.5 变址继电器运算结果脉冲化

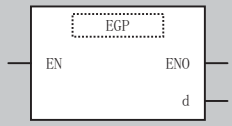
EGP, EGF

Basic | High performance | Universal | L CPU

EGP
EGF

EGP: 执行条件 : ↑
EGF: 执行条件 : ↓

结构体梯形图



ST

ENO:= EGP (EN ,d);

[ ] 中放入下述指令。

EGP
EGF

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: 存储运算结果的变址继电器编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 V
	位	字		位	字				
ⓓ	–								○

功 能

EGP

(1) 将 EGP 指令之前的运算结果存储到变址继电器 (V) 中。

(2) 在 EGP 指令之前的运算结果的上升沿时 (OFF → ON)，变为 ON(导通状态)。
在 EGP 指令之前的运算结果为上升沿以外 (ON → ON、ON → OFF、OFF → OFF) 的情况下，变为 OFF(非导通状态)。

(3) 对于 EGP 指令，在执行子程序及 FOR ～ NEXT 之间进行了变址修饰的程序的脉冲运算时使用。

(4) 对于 EGP 指令，可以作为 AND 处理使用。

EGF

(1) 将 EGF 指令之前的运算结果存储到变址继电器 (V) 中。

(2) EGF 指令之前的运算结果的下降沿时 (ON → OFF)，变为 ON(导通状态)。
在 EGF 指令之前的运算结果为下降沿以外 (OFF → ON、ON → ON、OFF → OFF) 的情况下，变为 OFF(非导通状态)。

(3) 对于 EGF 指令，在执行子程序及 FOR ～ NEXT 之间进行了变址修饰的程序的脉冲运算时使用。

(4) 对于 EGF 指令，可以作为 AND 处理使用。

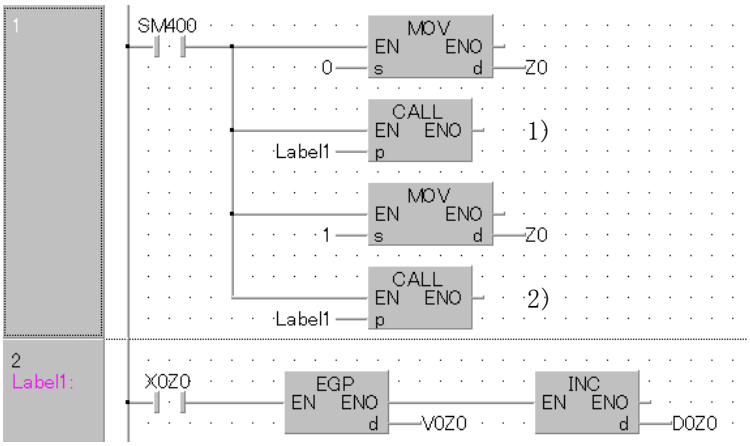
出 错

不存在 EGP、EGF 指令相关的运算出错。

程序示例

以下为使用了 EGP 指令的程序。

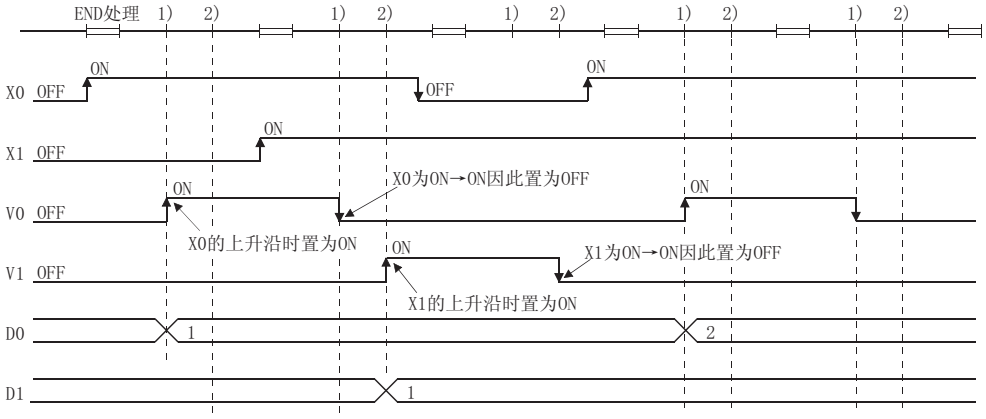
[结构体梯形图]



[ST]

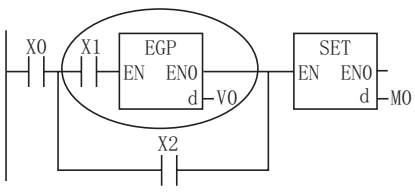
```
INC(EGP(X0Z0,V0Z0),D0Z0);
```

[动作]



要 点

1. 对于 EGP、EGF 指令，由于是以 EGP、EGF 指令之前的运算结果执行动作，因此应与 AND(参阅 5.1.1 项) 在同一位置处使用。
   对于 EGP、EGF 指令，不能在 LD、OR 的位置处使用。
2. 对于 EGP、EGF 指令，不能在下述梯形图块的位置处使用。



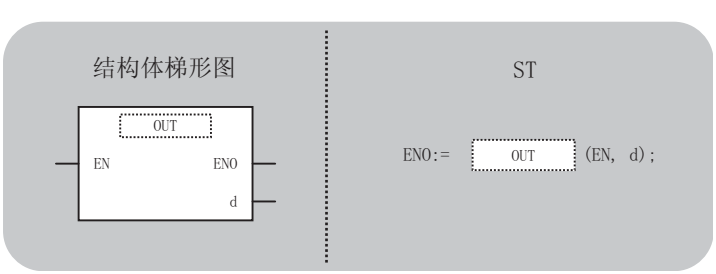
5.3 输出指令

5.3.1 输出（定时器、计数器、报警器除外）

OUT

Basic | High performance | Universal | L CPU

OUT



□ 中放入下述指令。

OUT

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: ON/OFF 的软元件编号 : ANY_SIMPLE

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
ⓓ	○（除 T、C、F 以外）		○				-		○

功 能

将 OUT 指令之前的运算结果输出到指定的软元件。

(a) 使用位软元件时

运算结果	线圈
OFF	OFF
ON	ON

(b) 字软元件的位指定时

运算结果	指定位
OFF	0
ON	1

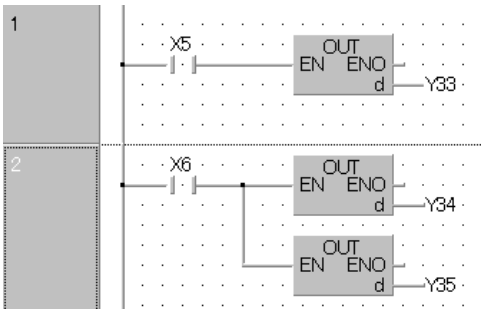
出 错

不存在 OUT 指令相关的运算出错。

程序示例

(1) 使用位软元件时

[结构体梯形图]

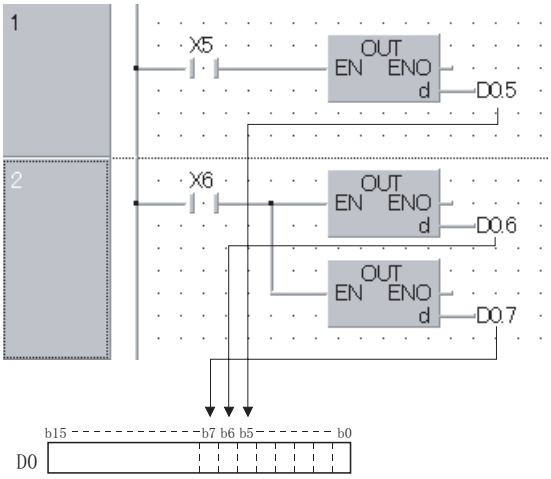


[ST]

```
OUT(X5,Y33);
IF X6 THEN
     OUT(TRUE,Y34);
     OUT(TRUE,Y35);
END_IF;
```

(2) 字软元件的位指定时

[结构体梯形图]



[ST]

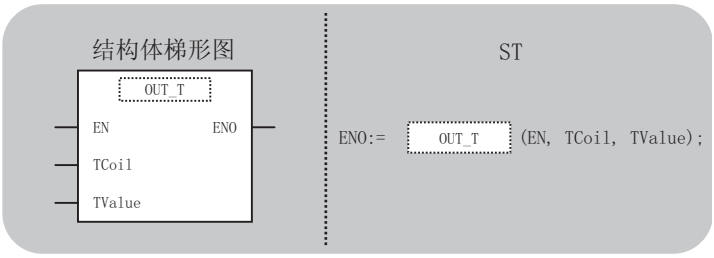
```
OUT(X5,D0.5);
IF X6 THEN
     OUT(TRUE,D0.6);
     OUT(TRUE,D0.7);
END_IF;
```

5.3.2 定时器

OUT_T, OUTH_T

Basic | High performance | Universal | L CPU

OUT_T
OUTH_T



[ ] 中放入下述指令。

OUT_T
OUTH_T

输入自变量， EN: 执行条件 ：位
TCoil: 定时器编号 ：位
TValue: 定时器的设置值 ：ANY16
输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K	其它
	位	字		位	字				
TCoil	○（仅 T）	－	－	－	－		－	－	－
TValue	－	○（除 T、C 以外）	○	－	○		－	○	－

功 能

(1) OUT(H)_T 指令之前的运算结果为 ON 时，将定时器的线圈置为 ON 并进行计测直至达到设置值为止，时间到（计数值≧设置值）时，触点的情况如下所示。

常开触点	定时器导通
常闭触点	定时器非导通

(2) OUT(H)_T 指令之前的运算结果发生了 ON → OFF 的变化时的情况如下所示。

定时器的种类	定时器线圈	定时器的当前值	时间到之前		时间到之后	
			常开触点	常闭触点	常开触点	常闭触点
低速定时器	OFF	0	非导通	导通	非导通	导通
高速定时器						
低速累计定时器	OFF	保持当前值	非导通	导通	导通	非导通
高速累计定时器						

(3) 时间到之后，通过 RST 指令进行累计定时器当前值的清除及触点的 OFF。

(4) 设置值不能设置为负数 (-32768 ～ -1)。
设置值为 0 时，执行 OUT(H)_T 指令时变为时间到。

(5) 执行 OUT(H)_T 指令时，进行下述处理。

• OUT(H)_T T□的线圈的 ON/OFF

• OUT(H)_T T□的触点的 ON/OFF

• OUT(H)_T T□的当前值的变更
OUT(H)_T 指令为 ON 过程中如果通过 JMP 指令等跳过 OUT(H)_T 指令，则不进行当前值的更新及触点的 ON/OFF。
此外，将同一 OUT(H)_T 指令在同一个扫描内执行了 2 次以上的情况下，将按执行的次数对当前值进行更新。

(6) 定时器的线圈 / 触点的变址修饰只能通过 Z0、Z1 进行。
对于定时器的设置值没有变址修饰方面的限制。

备 注

关于定时器的时限
时限的设置是在可编程控制器参数的可编程控制器系统设置中进行。

定时器的种类	基本型 QCPU、高性能型 QCPU		通用型 QCPU、LCPU	
	设置范围	设置单位	设置范围	设置单位
低速定时器	1ms ～ 1000ms （默认： 100ms）	1ms	1ms ～ 1000ms （默认： 100ms）	1ms
低速累计定时器	0.1ms ～ 100ms （默认： 10ms）	0.1ms	0.01ms ～ 100.0ms （默认： 10.0ms）	0.01ms

关于定时器的计数方法，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

不存在 OUT(H)_T 指令相关的运算出错。

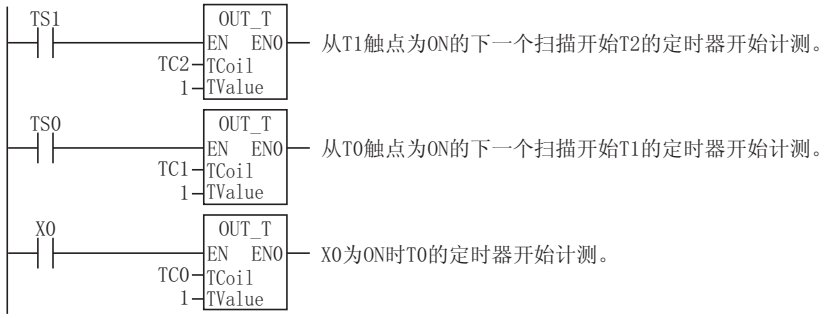
注意事项

(1) 创建通过定时器的触点进行其它定时器计测的程序时，应从后计测的定时器开始按顺序进行编程。
下述情况下，如果按计测顺序进行编程，所有的定时器将在同一个扫描中 ON。

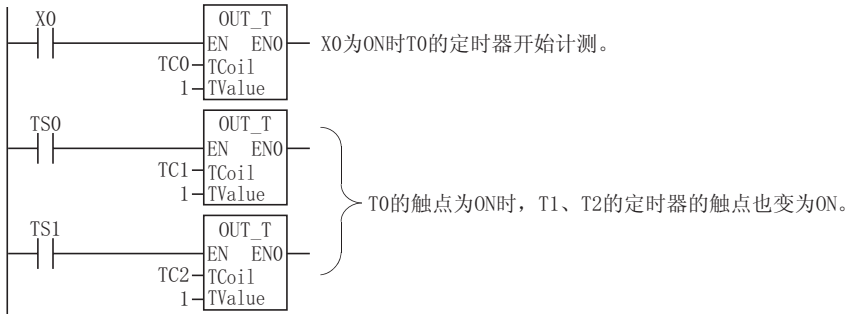
• 设置值短于扫描时间时

• 设置值为 1 时

示例　对 T0 ～ T2 的定时器按照从后计测的定时器开始的顺序进行了编程的情况下



示例　对 T0 ～ T2 的定时器按计测顺序进行了编程的情况下



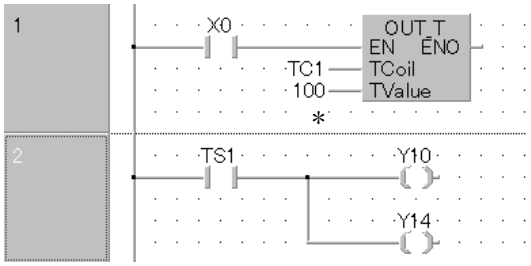
(2) 在程序中使用定时器的软元件时，需要根据使用位置按下述分类分别使用。
输入了 T、ST 的情况下，处理为 TN、STN。

- 用于触点时 : TS、STS
- 用于线圈时 : TC、STC
- 用于当前值时 : TN、STN

程序示例

(1) 以下为 X0 变为 ON 且经过了 10 秒后将 Y10、Y14 置为 ON 的程序。

[结构体梯形图]



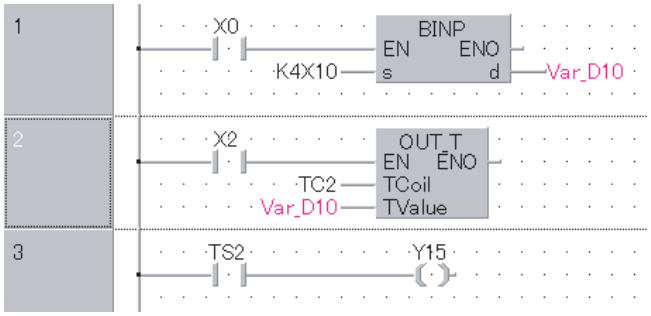
[ST]

```
OUT_T(X0,TC1,100);
OUT(TS1,Y10);
OUT(TS1,Y14);
```

*2 :　对于低速定时器的设置值，表示默认值的时限 (100ms) 的情况。

(2) 以下为将 X10 ～ X1F 的 BCD 数据设置为定时器的设置值的程序。

[结构体梯形图]

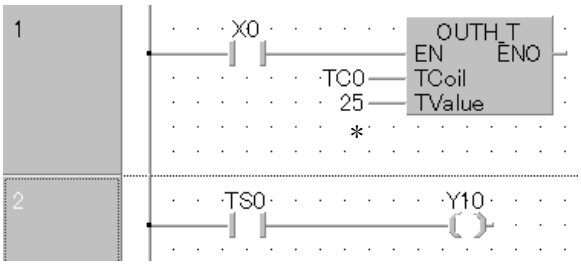


[ST]

```
BINP(X0,K4X10,Var_D10);
OUT_T(X2,TC2,Var_D10);
OUT(TS2,Y15);
```

(3) 以下为 X0 变为 ON 且经过了 250ms 后将 Y10 置为 ON 的程序。

[结构体梯形图]



[ST]

```
OUTH_T(X0,TC0,25);
OUT(TS0,Y10);
```

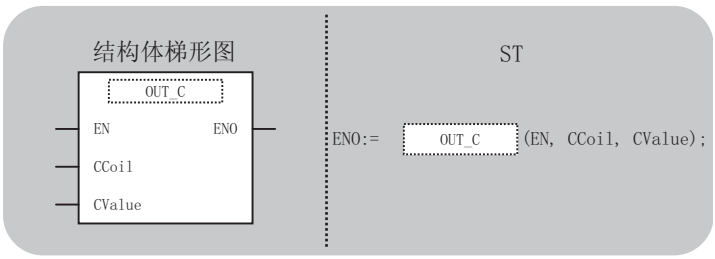
*3 ： 对于高速定时器的设置值，表示默认值的时限（10ms）的情况。

5.3.3 计数器

OUT_C

Basic | High performance | Universal | L CPU

OUT_C



[ ] 中放入下述指令。

OUT_C

输入自变量， EN: 执行条件 ：位
CCoil: 计数器编号 ：位
CValue: 计数器的设置值 ：ANY16
输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 K	其它
	位	字		位	字				
CCoil	○(仅CC)	–	–	–	–		–	–	–
CValue	–	○ (除T、C以外)	○	–	○		–	○	–

功 能

(1) OUT_C 指令之前的运算结果发生了 OFF → ON 的变化时，对当前值（计数值）进行 +1，计数到（当前值 = 设置值）时，触点的情况如下所示。

常开触点	导通
常闭触点	非导通

(2) 运算结果保持为 ON 不变时不进行计数。（ 计数输入无需进行脉冲化。）

(3) 计数到后，在执行 RST 指令之前计数值及触点的状态不变化。

(4) 设置值不能设置为负数 (-32768 ～ -1)。
此外，设置值为 0 时，与 1 时的处理相同。

(5) 计数器的线圈 / 触点的变址修饰只能使用 Z0、Z1。
对于计数器的设置值，无变址修饰方面的限制。

(6) 在程序中使用计数器的软元件时，需要根据使用位置按下述分类分别使用。
输入了 C 时，作为 CN 处理。

- 用于触点时 : CS
- 用于线圈时 : CC
- 用于当前值时 : CN

备 注

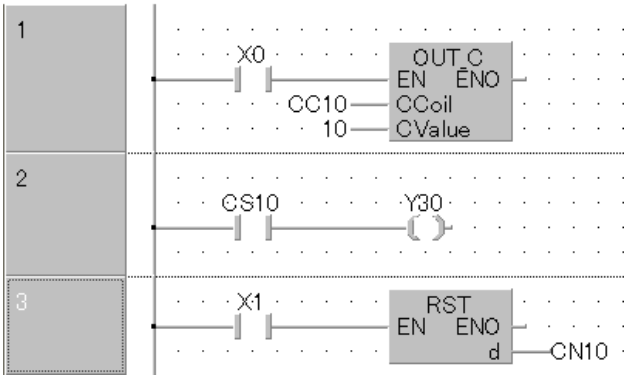
关于计数器的计数方法，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

不存在 OUT_C 指令相关的运算出错。

程序示例

(1) 以下为 X0 为 10 次 ON 之后将 Y30 置为 ON，X1 变为 ON 时对计数器进行复位的程序。
[结构体梯形图]

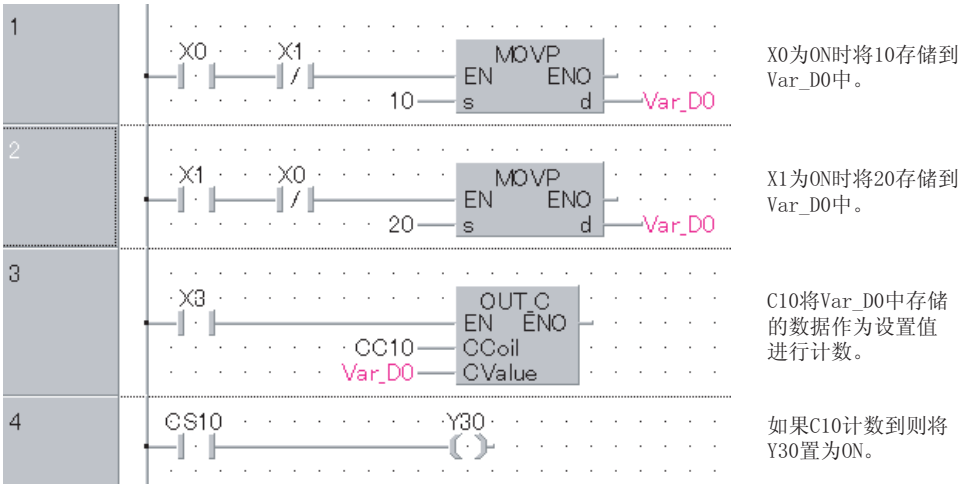


[ST]

```
OUT_C(X0,CC10,10);
OUT(CS10,Y30);
RST(X1,CN10);
```

(2) 以下为 X0 为 ON 时将 C10 的设置值设置为 10，X1 为 ON 时将 C10 的设置值设置为 20 的程序。

[结构体梯形图]



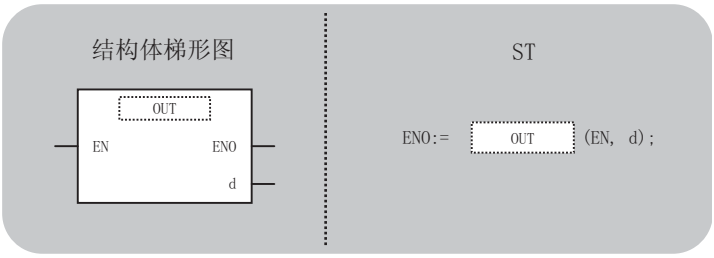

```
[ST]
IF X0 AND NOT(X1) THEN
    MOVP(TRUE, 10, Var_D0);
END_IF;
IF NOT(X0) AND X1 THEN
    MOVP(TRUE, 20, Var_D0);
END_IF;
OUT_C(X3,CC10,Var_D0);
OUT(CS10,Y30);
```

5.3.4 报警器输出

OUT

Basic | High performance | Universal | L CPU

OUT



[ ] 中放入下述指令。

OUT

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: 变为 ON 的报警器编号 : ANY_SIMPLE

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	○ (仅 F)		–						

功 能

(1) 将 OUT 指令之前的运算结果输出到指定的报警器。

(2) 报警器 (F) 变为 ON 的情况如下所示。

- “USER” / “ERR.” LED 亮灯。
- 将变为 ON 的报警器编号 (F 编号) 存储到特殊寄存器 (SD64 ～ SD79) 中。
- 对 SD63 的内容进行 +1。

(3) SD63 的内容为 16 的情况下 (报警器已有 16 个处于 ON 状态)，即使有新的报警器 ON，变为 ON 的报警器编号也不被存储到 SD64 ～ SD79 中。

(4) 通过 OUT 指令将报警器置为 OFF 时的情况如下所示。

线圈变为 OFF，但 “USER” / “ERR.” LED 的状态、SD63 ～ SD79 的内容无变化。
希望使 “USER” / “ERR.” LED 熄灯、将通过 OUT 指令置为 OFF 的报警器从 SD63 ～ SD79 中删除的情况下，应使用 RST 指令。

出 错

不存在 OUT 指令相关的运算出错。

备 注

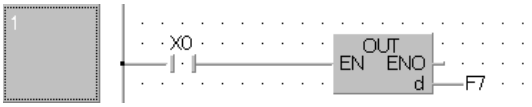
1. 关于报警器的有关内容，请参阅所使用的 CPU 模块的用户手册（功能解解说 / 程序基础篇）。
2. CPU 模块前面的 LED 显示器或 “USER” LED 的 CPU 模块类型如下所示。

LED 类型	CPU 模块型号
“USER” LED	高性能型 QCPU、通用型 QCPU、LCPU
“ERR.” LED	基本型 QCPU

程序示例

以下为 X0 为 ON 时将 F7 置为 ON，将 7 存储到 SD64 ～ SD79 中的程序。

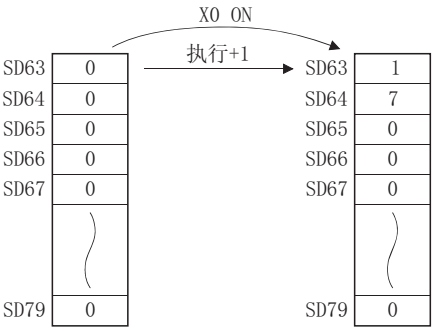
[结构体梯形图]



[ST]

```
OUT(X0,F7);
```

[动作]



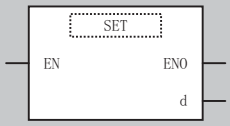
5.3.5 软元件的设置（报警器除外）

SET

Basic | High performance | Universal | L CPU

SET

结构体梯形图



ST

ENO:= SET (EN, d);

□ 中放入下述指令。

SET

输入自变量， EN: 执行条件 ：位
输出自变量， ENO: 执行结果 ：位
d: 设置（ON）的位软元件编号 / 字软元件的位指定 ：位

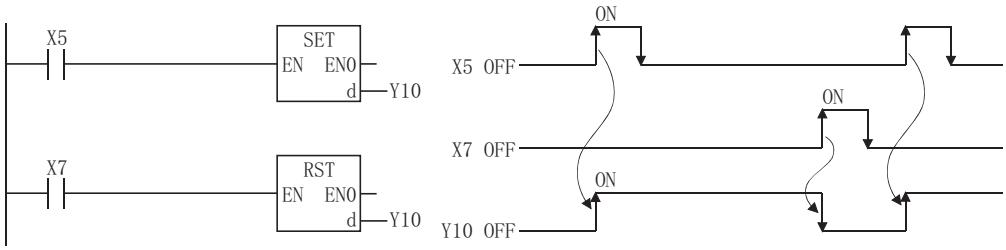
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				BL, DY
ⓓ	○	○（除 T、C 以外）		○			－		○

功 能

（1）执行指令为 ON 时指定软元件的情况如下所示。

软元件	软元件的状态
位软元件	将线圈、触点置为 ON。
字软元件的位指定时	将指定位置为 1。

（2）对于被置为 ON 的软元件，即使执行指令变为 OFF 也仍将保持为 ON 状态不变。
对于通过 SET 指令置为 ON 的软元件，可以通过 RST 指令置为 OFF。



（3）执行指令为 OFF 的情况下，软元件的状态不变化。

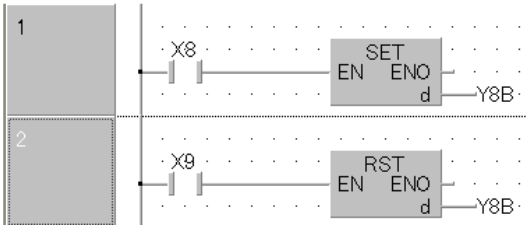
出 错

不存在 SET 指令相关的运算出错。

程序示例

(1) 以下为 X8 变为 ON 时对 Y8B 进行设置 (ON)，将 X9 置为 ON 时对 Y8B 进行复位 (OFF) 的程序。

[结构体梯形图]

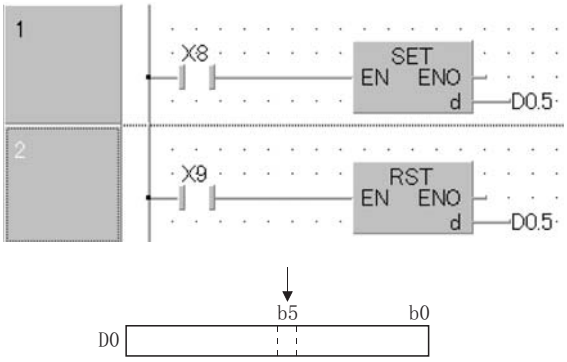


[ST]

```
SET(X8,Y8B);
RST(X9,Y8B);
```

(2) 以下为 X8 变为 ON 时将 D0 的位 5(b5) 置为 1，X9 变为 ON 时将 D0 的位 5(b5) 置为 0 的程序。

[结构体梯形图]



[ST]

```
SET(X8,D0.5);
RST(X9,D0.5);
```

备 注

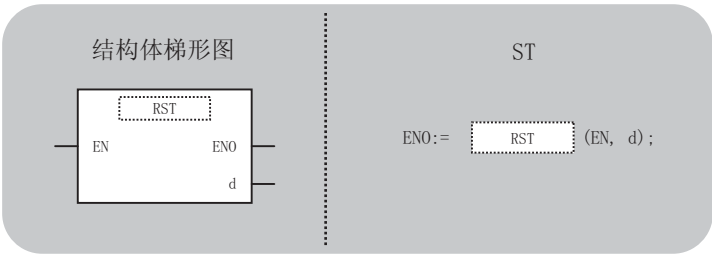
在软元件中使用 X 的情况下，应使用实际输入中未使用的软元件编号。如果使用与实际输入软元件相同的编号，则实际输入的数据将被 SET 指令中指定的输入 X 所覆盖。

5.3.6 软元件的复位（报警器除外）

RST

Basic | High performance | Universal | L CPU

RST



[ ] 中放入下述指令。

RST

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: 复位的位软元件编号 / 字软元件的位指定 : ANY_SIMPLE

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DY
ⓓ	○							-	○

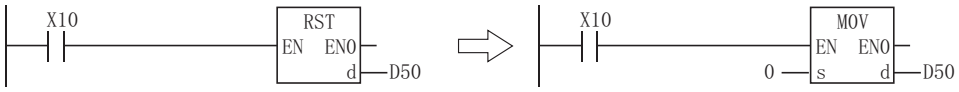
功 能

(1) 执行指令为 ON 时指定软元件的情况如下所示。

软元件	软元件的状态
位软元件	将线圈、触点置为 OFF。
定时器、计数器	将当前值置为 0，将线圈、触点置为 OFF。
字软元件的位指定时	将指定位置为 0。
定时器、计数器以外的字软元件	将内容置为 0。

(2) 执行指令为 OFF 的情况下，软元件的状态不变化。

(3) 通过 RST 指令指定字软元件时的功能与以下梯形图相同。



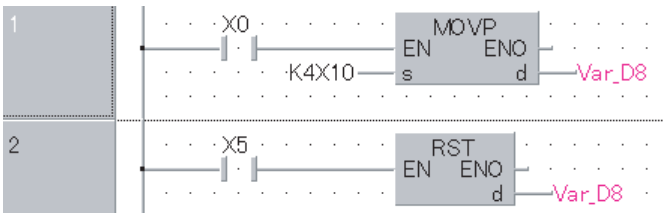
出 错

不存在 RST 指令相关的运算出错。

程序示例

(1) 以下为将数据寄存器的内容置为 0 的程序。

[结构体梯形图]



X0为ON时将X10～X1F的内容存储到Var_D8中。

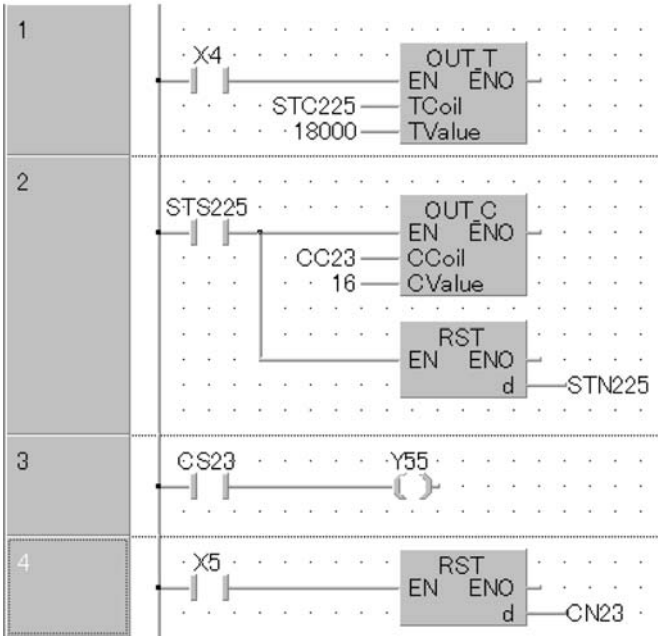
X5为ON时将Var_D8的内容置为0。

[ST]

```
MOVP(X0,K4X10,Var_D8);
RST(X5,Var_D8);
```

(2) 以下为对 100ms 累计定时器、计数器进行复位的程序。

[结构体梯形图]



ST225被设置为累计定时器时，X4的ON时间为30分时ST225将变为ON。

对ST225变为ON的次数进行计数。

ST225的触点变为ON时对ST225的线圈、触点、当前值进行复位。

C23为计数到时Y55将变为ON。

X5为ON时对C23进行复位。

```
[ST]
OUT_T(X4,STC225,18000);
IF STS225 THEN
     OUT_C(TRUE,CC23,16);
     RST(TRUE,STN225);
END_IF;
OUT(CS23,Y55);
RST(X5,CN23);
```

备 注

对定时器、计数器软元件进行复位时，输出自变量ⓓ中只能指定线圈 (TC、STC、CC) 或当前值 (TN、STN、CN)。

不能指定触点 (TS、STS、CS)。

5.3.7 报警器的设置、复位

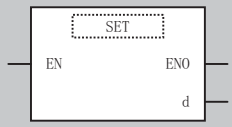
SET, RST

Basic | High performance | Universal | L CPU

SET
RST

执行条件 : ↑

结构体梯形图



ST

ENO:= SET (EN, d);

[ ] 中放入下述指令。

SET
RST

输入自变量, EN: 执行条件 : 位
输出自变量, ENO: 执行结果 : 位
d: 设置的报警器编号 : 位
复位的报警器编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	○(仅F)		-						

功 能

SET

(1) 执行指令为 ON 时，将ⓓ 中指定的报警器置为 ON。

(2) 将报警器 (F) 置为 ON 时的情况如下所示。

- “USER”LED 亮灯。*1
- 将变为 ON 的报警器编号 (F 编号) 存储到特殊寄存器 (SD64 ～ SD79) 中。
- 将 SD63 的内容进行 +1。

*1 : 基本型 QCPU 的情况下，“ERR.”LED 将亮灯。

(3) SD63 的内容为 16 时 (报警器已有 16 个处于 ON 状态)，即使有新的报警变为 ON，变为 ON 的报警器编号也不被存储到 SD64 ～ SD79 中。

RST

(1) 执行指令为 ON 时，将@ 中指定的报警器置为 OFF。

(2) 变为 OFF 的报警器编号（F 编号）将从特殊寄存器（SD64 ～ SD79）被删除，SD63 的内容将被进行 -1。

备 注

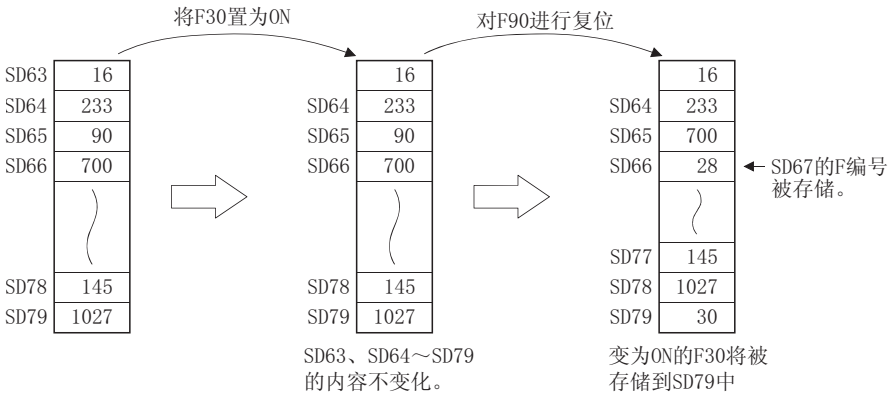
关于报警器的有关内容，请参阅所使用的 CPU 模块的用户手册（功能解解说 / 程序基础篇）。

(3) SD63 的内容为 16 时，通过 RST 指令可以从 SD64 ～ SD79 中将报警器编号删除。此外，如果 SD64 ～ SD79 中登录的编号的报警器变为 ON，将该编号重新登录。

- 如果 SD64 ～ SD79 的报警器编号全部变为 OFF，则 CPU 模块前面的 LED 显示或 “USER” LED 将熄灯。*2

*2: 基本型 QCPU 的情况下，“ERR.” LED 将亮灯。

[SD63 为 16 时的动作]



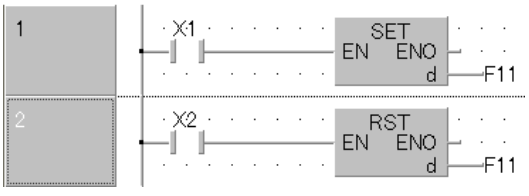
出 错

不存在 SET、RST 指令相关的运算出错。

程序示例

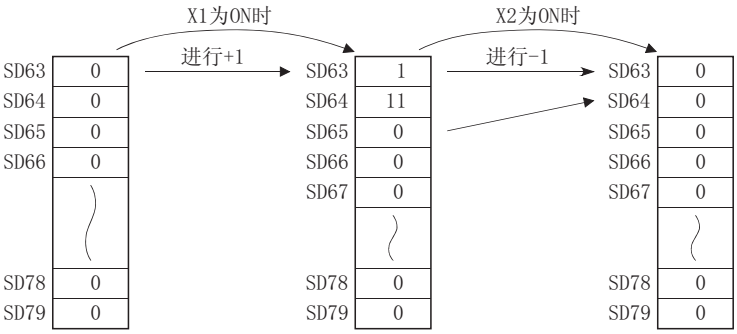
以下程序为 X1 变为 ON 时，将报警器 F11 置为 ON，在特殊寄存器（SD64 ～ SD79）中存储 11。此外，X2 变为 ON 时对报警器 F11 进行复位，将特殊寄存器（SD64 ～ SD79）内的 11 删除的程序。

[结构体梯形图]



[ST]
SET(X1,F11);
RST(X2,F11);

[动作]



5.3.8 上升沿、下降沿输出

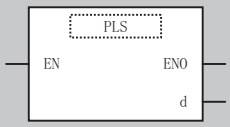
PLS, PLF

Basic | High performance | Universal | L CPU

PLS
PLF

PLS：执行条件 ：↑
PLF：执行条件 ：↓

结构体梯形图



ST

```
ENO:= PLS (EN, d);
```

[ ] 中放入下述指令。

PLS
PLF

输入自变量， EN: 执行条件 ：位
输出自变量， ENO: 执行结果 ：位
d: 脉冲化的软元件 ：位

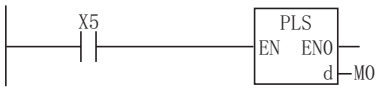
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				DY
ⓓ	○						–		○

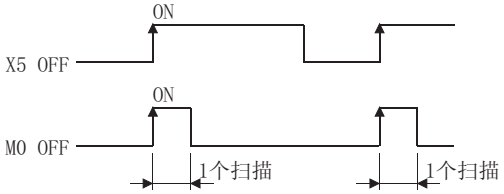
功 能

PLS

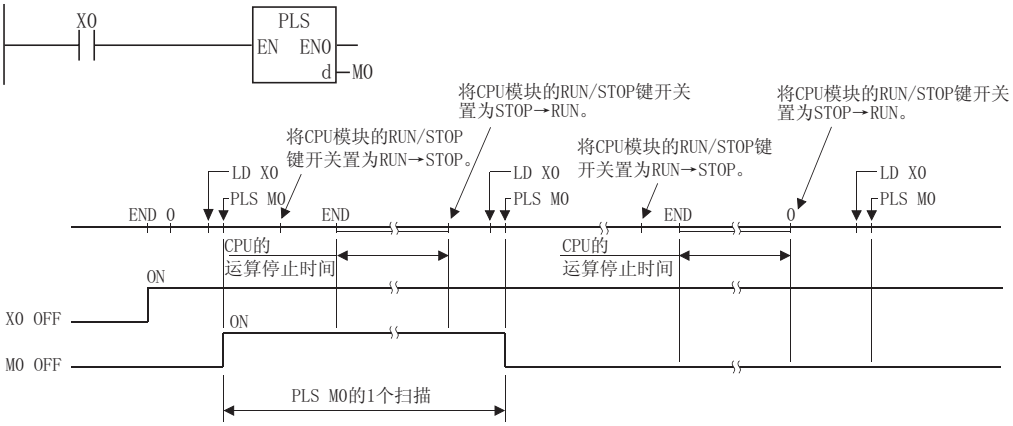
(1) 执行指令 OFF → ON 时将指定软元件置为 ON，除执行指令的 OFF → ON 以外（ON → ON、ON → OFF、OFF → OFF）时将其置为 OFF。

1 个扫描ⓓ中指定软元件的 PLS 指令为 1 个时，指定软元件将 1 个扫描 ON。
关于将同一软元件的 PLS 指令在 1 个扫描执行了多次时的动作，请参阅 3.4 节。





(2) 执行 PLS 指令后进行 RUN → STOP 后，即使再次 RUN 也不能执行 PLS 指令。



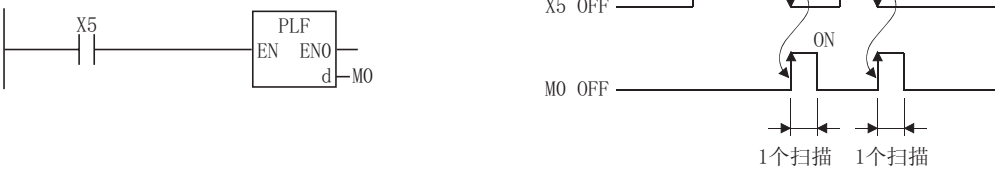
(3) 执行指令中指定锁存继电器 (L) 后，在锁存继电器为 ON 状态下如果将电源 OFF → ON，执行指令将在第 1 个扫描中变为 OFF → ON，执行 PLS 指令后指定软元件将变为 ON。

对于电源 ON 后第 1 个扫描中变为 ON 的软元件，通过下一个 PLS 指令置为 OFF。

PLF

(1) 执行指令 ON → OFF 时将指定软元件置为 ON，执行指令为 ON → OFF 以外 (OFF → OFF、OFF → ON、ON → ON) 时将其置为 OFF。

在 1 个扫描ⓓ中指定软元件的 PLF 指令为 1 个的情况下，指定软元件将 1 个扫描 ON。
将同一软元件的 PLF 指令在 1 个扫描执行了多次的情况下，其动作情况请参阅 3.4 节。



(2) 执行 PLF 指令后进行 RUN → STOP 后，即使再次 RUN 也不能执行 PLF 指令。

要 点

将 PLS、PLF 指令通过 CJ 指令进行了跳转，对执行的子程序未通过 CALL 指令进行调用的情况下，ⓓ中指定的软元件有可能 1 个扫描以上 ON，应加以注意。

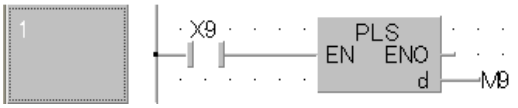
出 错

不存在 PLS、PLF 指令相关的运算出错。

程序示例

(1) 以下为 X9 变为 ON 时，执行 PLS 指令的程序。

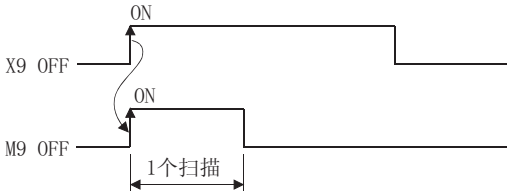
[结构体梯形图]



[ST]

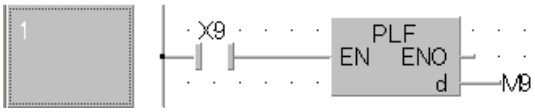
```
PLS(X9,M9);
```

[时序图]



(2) 以下为 X9 变为 OFF 时，执行 PLF 指令的程序。

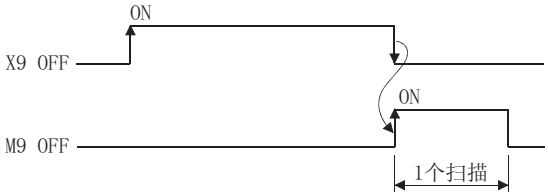
[结构体梯形图]



[ST]

```
PLF(X9,M9);
```

[时序图]



5.3.9 位软元件输出取反

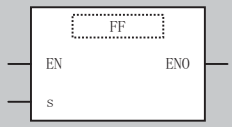
FF

Basic | High performance | Universal | L CPU

FF

执行条件 : ↑

结构体梯形图



ST

ENO:= FF (EN, s);

中放入下述指令。

FF

输入自变量， EN: 执行条件 : 位
s: 取反的软元件编号 : 位
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
Ⓢ			○				-		○

功 能

执行指令的 OFF → ON 时，对Ⓢ中指定的软元件状态进行取反。

软元件	软元件的状态	
	执行 FF 前	执行 FF 后
位软元件	OFF	ON
	ON	OFF
字软元件的位指定	0	1
	1	0

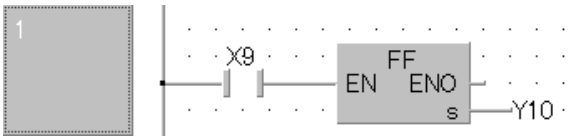
出 错

不存在 FF 指令相关的运算出错。

程序示例

(1) 以下为 X9 变为 ON 时，对 Y10 的输出进行取反的程序。

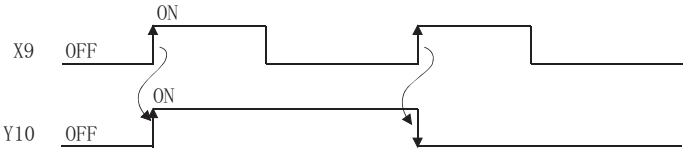
[结构体梯形图]



[ST]

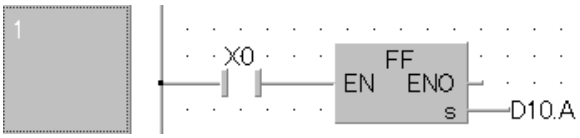
```
FF(X9,Y10);
```

[时序图]



(2) 以下为 X0 变为 ON 时，对 D10 的 b10(位 10) 进行取反的程序。

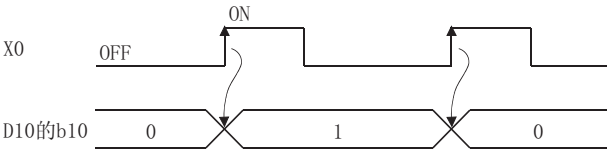
[结构体梯形图]



[ST]

```
FF(X0,D10.A);
```

[时序图]



5.3.10 直接输出的脉冲化

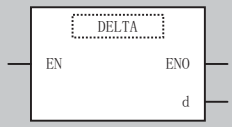
DELTA

Basic | High performance | Universal | L CPU

DELTA(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= DELTA (EN, d);

中放入下述指令。

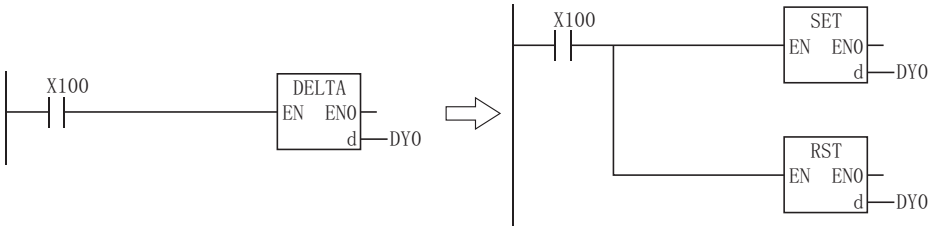
DELTA DELTAP

输入自变量， EN: 执行条件 :位
输出自变量， ENO: 执行结果 :位
d: 脉冲化的位 :位

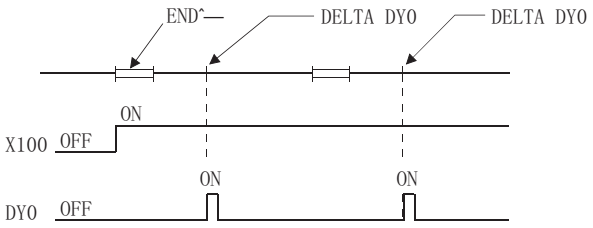
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
ⓓ			-						○

功 能

(1) 对ⓓ中指定的直接访问输出 (DY) 进行脉冲输出。
对 DELTA DY0 进行了指定的情况下，与下图所示使用了 SET/RST 指令的梯形图的动作相同。



[动作]



(2) DELTAP 指令用于至智能功能模块的上升沿执行指令。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

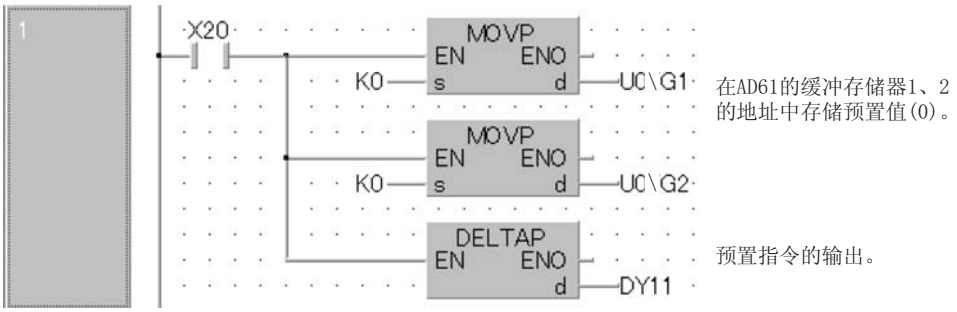
• ⓓ 中指定的直接访问输出编号超出了 CPU 模块的输出范围时。

（出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，对主基板的插槽 0 上安装的 AD61 的 CH1 进行预置的程序。

[结构体梯形图]




```
[ST]
IF X20 THEN
    MOVP(TRUE,K0,U0¥G1);
    MOVP(TRUE,K0,U0¥G2);
    DELTAP(TRUE,DY11);
END_IF;
```

5.4 移动指令

5.4.1 位软元件移动

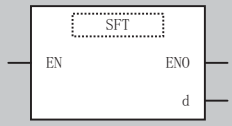
SFT

Basic | High performance | Universal | L CPU

SFT(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SFT (EN, d);

中放入下述指令。

SFT　　SFTP

输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位
　　d:　移动的软元件编号　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 DY
	位	字		位	字				
ⓓ	○（除 T、C 以外）						-		○

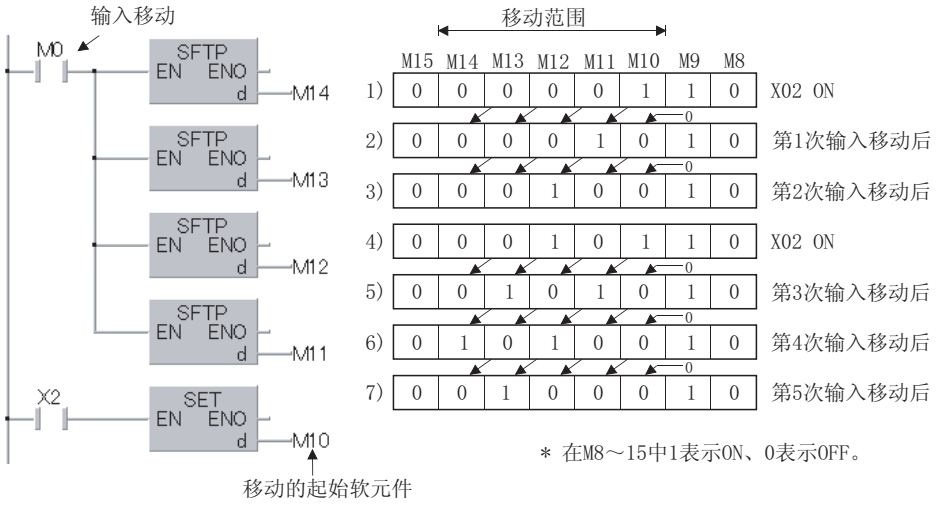
功 能

(1) 位软元件的情况

(a) 对于ⓓ中指定的软元件，将小 1 号的软元件的 ON/OFF 状态移动至ⓓ中指定的软元件处，将小 1 号的软元件置为 OFF。
例如，通过 SFT 指令指定 M11 的情况下，执行 SFT 指令时将 M10 的 ON/OFF 移动至 M11 处，将 M10 置为 OFF。

(b) 对于移动的起始软元件应通过 SET 指令置为 ON。

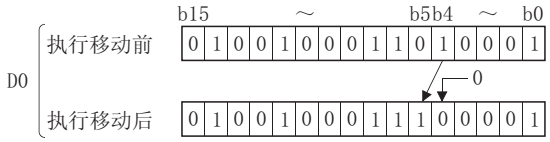
(c) 连续使用 SFT、SFTP 的情况下，从软元件编号的大号开始进行编程。



(2) 字软元件的位指定的情况

(a) 对于ⓓ中指定的软元件的位，将低 1 位的 1/0 状态移动至ⓓ中指定的位处，将低 1 位置为 0。

例如，通过 SFT 指令指定 D0.5〔D0 的位 5(b5)〕的情况下，在执行 SFT 指令时将 D0 的 b4 的 1/0 移动至 b5 处，将 b4 置为 0。



出 错

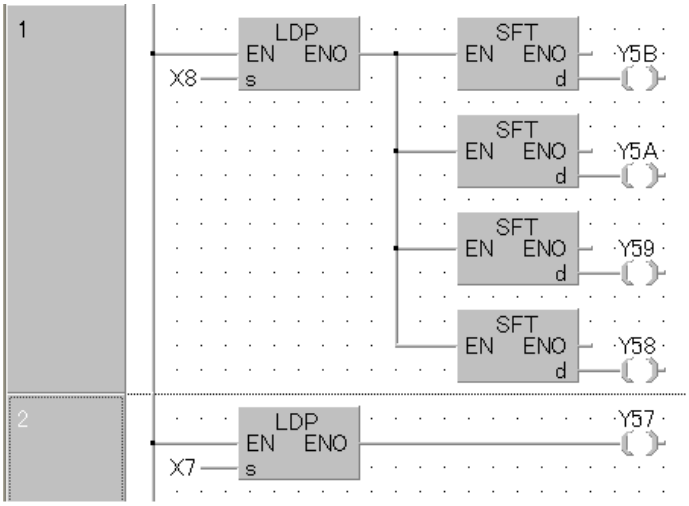
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 的情况下） （出错代码： 4101）

程序示例

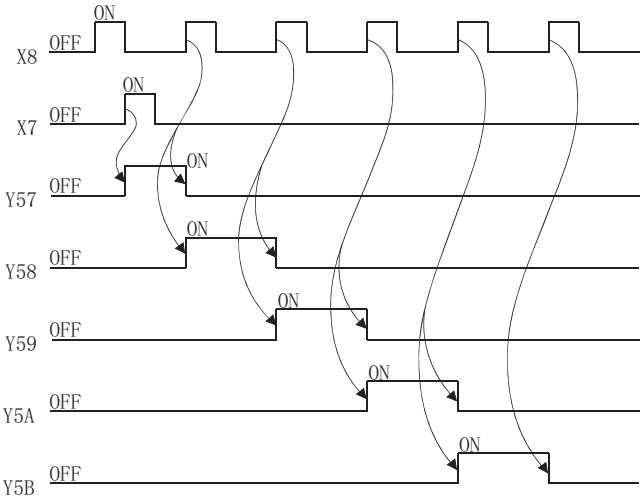
以下为 X8 变为 ON 时，对 Y57 ～ Y5B 进行移动的程序。

[结构体梯形图]




```
[ST]
IF LDP(TRUE,X8) THEN
     SFT(TRUE,Y5B);
     SFT(TRUE,Y5A);
     SFT(TRUE,Y59);
     SFT(TRUE,Y58);
END_IF;
IF LDP(TRUE,X7) THEN
     OUT(TRUE,Y57);
END_IF;
```

[时序图]



5.5 主控指令

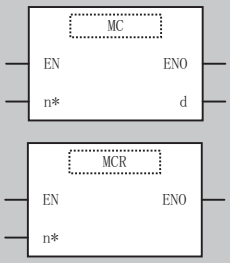
5.5.1 主控的设置、复位

MC, MCR

Basic | High performance | Universal | L CPU

MC
MCR

结构体梯形图



ST

ENO:= MC (EN, n*, d);

ENO:= MCR (EN, n*);

中放入下述指令。

MC
MCR

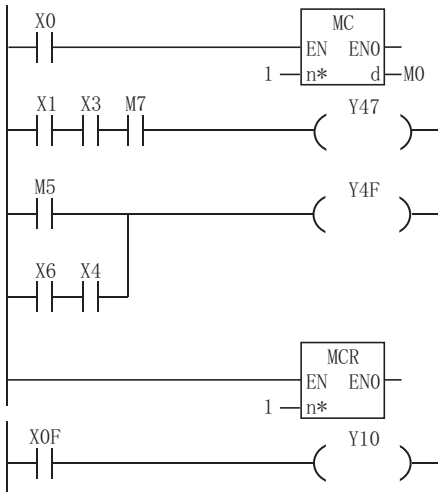
输入自变量， EN: 执行条件 ：位
n*: 嵌套 (N0 ～ N14) ：ANY16
输出自变量， ENO: 执行结果 ：位
d: 置为 ON 的软元件编号 (MC 时) ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它	
	位	字		位	字				N	DY
n*	－						－		○	－
ⓓ	○						－		－	○

功 能

主控指令是指，通过对梯形图的公共母线进行开闭，以创建高效切换梯形图的顺控程序的指令。

使用了主控的梯形图如下所示。



MC

(1) 通过主控的开始，MC 指令的执行指令变为 ON 的情况下，MC 指令与 MCR 命令之间的运算结果如上图的指令（梯形图）所示。

MC 的执行指令变为 OFF 的情况下，MC 指令与 MCR 指令之间的运算结果情况如下所示。

软元件	软元件的状态
高速定时器 低速定时器	计数值变为 0，线圈、触点均变为 OFF。
高速累计定时器 低速累计定时器 计数器	线圈将变为 OFF，但计数值、触点均保持为当前的状态不变。
OUT 指令中的软元件	全部变为 OFF。
SET,RST SFT 指令中 基本、应用 } 指令中的软元件	保持当前的状态。

(2) MC 指令变为 OFF 的情况下 MC 指令与 MCR 指令之间的指令也仍然执行，因此扫描时间不会变短。

要 点

在使用了主控的梯形图中，存在有无需触点指令的指令（FOR ～ NEXT、EI、DI 指令等）的情况下，CPU 模块与 MC 指令的执行指令无关将执行该指令。

(3) 对于 MC 指令，通过对ⓓ软元件进行更改，可以将同一嵌套编号多次使用。

(4) MC 指令变为 ON 时，ⓓ中指定的软元件的线圈将变为 ON。

此外，由于通过 OUT 指令等使用同一个软元件时将变为双线圈，因此不要在其它指令中使用ⓓ中指定的软元件。

MCR

(1) 是主控的解除指令，表示主控的范围结束。

(2) 在 MCR 指令的前面不要附加触点。

(3) 应将 MC 指令与 MCR 指令设置为相同的嵌套编号使用。
但是，嵌套结构集中在一个位置时，通过一个最小嵌套编号的 MCR 指令，可以结束所有的主控。
（参阅程序示例的 “嵌套结构时的注意事项”）

出 错

不存在 MC、MCR 指令相关的运算出错。

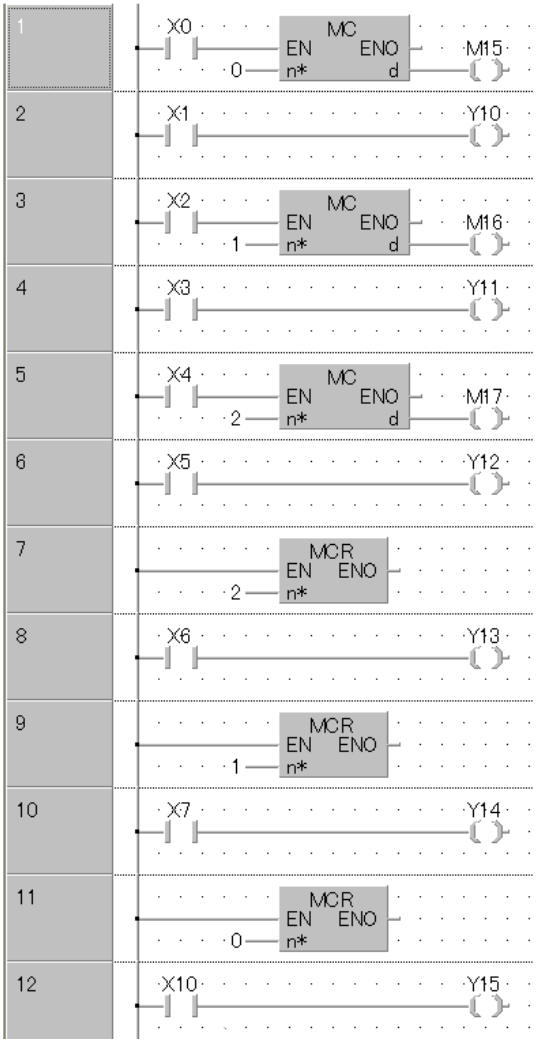
程序示例

对主控指令可以使用嵌套结构。将各个主控区间通过嵌套编号加以区分。可使用 0 ～ 14 个嵌套。

通过使用嵌套结构，可以创建对程序的执行条件依次加以限制的梯形图。

使用了嵌套结构的梯形图如下所示。

[结构体梯形图]



嵌套结构时的注意事项

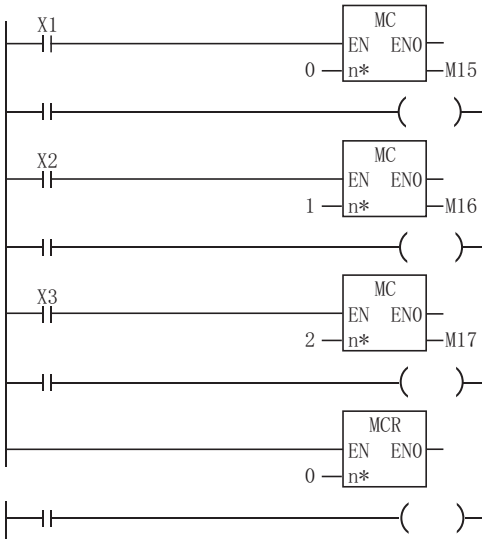
(1) 嵌套的可使用个数最多为 15 个 (0 ～ 14)。

使用嵌套的情况下，在 MC 指令中从嵌套编号的小号开始使用，对于 MCR 指令则从大号开始使用。

如果编号顺序弄反，则不能构成嵌套结构，CPU 模块将无法正常运算。

例如，在 MC 指令中将嵌套按 1 → 0 的顺序进行指定，在 MCR 指令中将嵌套按 1 → 0 顺序指定的情况下，纵母线将会交叉。因此，不能构成正常的主控梯形图。

(2) 嵌套结构集中在一个位置时，通过一个最小嵌套编号的 MCR 指令，可以结束所有的主控。



5.6 结束指令

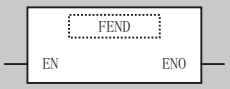
5.6.1 主程序结束

FEND

Basic | High performance | Universal | L CPU

FEND

结构体梯形图



ST

ENO:= FEND (EN);

[ ] 中放入下述指令。

FEND

输入自变量，EN: 执行条件 :位

输出自变量，ENO: 执行结果 :位

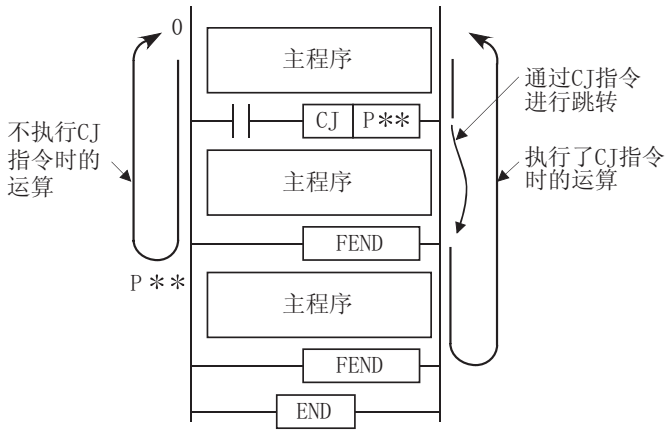
设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

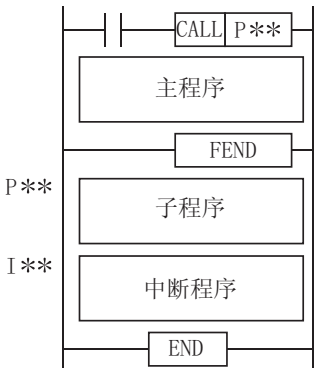
(1) 通过 CJ 指令等对顺控程序的运算进行分支的情况下，FEND 指令用于将主程序分割为子程序、中断程序。

(2) 如果执行 FEND 指令，CPU 模块将结束正在执行的程序。

(3) FEND 指令以后的顺控程序也可通过编程工具进行梯形图显示。
（编程工具显示 END 之前的梯形图。）



(a) 使用CJ指令时



(b) 有子程序、中断程序时

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 CALL 指令后，在执行 RET 指令之前执行了 FEND 指令时。 （出错代码： 4211）
- 执行 FOR 指令后，在执行 NEXT 指令之前执行了 FEND 指令时。 （出错代码： 4200）
- 在中断程序中，在执行 IRET 指令之前执行了 FEND 指令时。 （出错代码： 4221）
- 在 CHKCIR ～ CHKEND 指令内执行了 FEND 指令时。 （出错代码： 4230）
- 在 IX ～ IXEND 指令内执行了 FEND 指令时。 （出错代码： 4231）

5.7 其它指令

5.7.1 顺控程序停止

STOP

Basic | High performance | Universal | L CPU

STOP

结构体梯形图

ST

ENO:= STOP (EN);

[ ] 中放入下述指令。

STOP

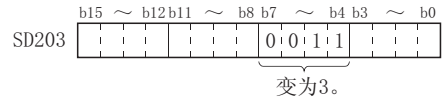
输入自变量，EN: 执行条件 :位
输出自变量，ENO: 执行结果 :位

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

(1) 执行指令变为 ON 时，对输出 Y 进行复位，停止 CPU 模块的运算。
（与将开关置为 STOP 侧的情况相同。）

(2) 如果执行 STOP 指令，特殊寄存器 SD203 的 b4 ～ b7 将变为 3。



(3) 执行 STOP 指令后如果希望重新开始 CPU 模块运算，则将开关置为 RUN → STOP 后，再次置为 RUN 位置。

出 错

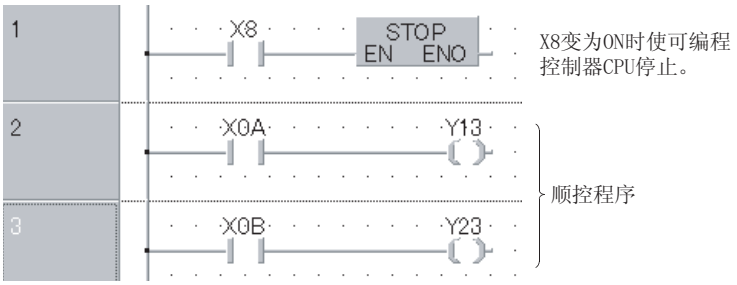
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 CALL 指令后，在执行 RET 指令之前执行了 STOP 指令时。 （出错代码： 4211)
- 执行 FOR 指令后，在执行 NEXT 指令之前执行了 STOP 指令时。 （出错代码： 4200)
- 在中断程序中，在执行 IRET 指令之前执行了 STOP 指令时。 （出错代码： 4221)
- 在 CHKCIR ～ CHKEND 指令内执行了 STOP 指令时。 （出错代码： 4230)
- 在 IX ～ IXEND 指令内执行了 STOP 指令时。 （出错代码： 4231)
- 在恒定周期执行类型程序中执行了 STOP 指令时。
  （通用型 QCPU、LCPU 时） （出错代码： 4223)

程序示例

以下为 X8 变为 ON 时停止 CPU 模块的程序。

[结构体梯形图]




```
[ST]
STOP(X8);
OUT(X0A,Y13);
OUT(X0B,Y23);
```

6 基本指令

6.1 比较运算指令

6.1.1 BIN16 位数据比较

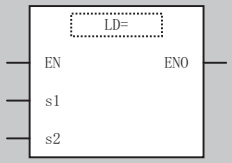
□=, □<>, □<=, □<, □>=, □>

Basic | High performance | Universal | L CPU

LD	(=)
AND	(<>)
OR	(<=)
	(<)
	(>=)
	(>)

□=	: =
□<>	: ≠
□<=	: ≤
□<	: <
□>=	: ≥
□>	: >

结构体梯形图



ST

不支持。

□ 中放入下述指令。

LD=	AND=	OR=
LD<>	AND<>	OR<>
LD<=	AND<=	OR<=
LD<	AND<	OR<
LD>=	AND>=	OR>=
LD>	AND>	OR>

输入自变量， EN: 执行条件 : 位
s1, s2: 比较数据或存储比较数据的软元件的起始编号 : ANY16
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	○								-
ⓢ2	○								-

功 能

(1) 将 ⓢ1 中指定的软元件的 BIN16 位数据与 ⓢ2 中指定的软元件的 BIN16 位数据通过常开触点进行比较运算。

(2) 各指令的比较运算结果如下所示。

□内指令符号	条件	比较运算结果	□内指令符号	条件	比较运算结果
□=	ⓢ1 =ⓢ2	导通状态	□=	ⓢ1 ≠ ⓢ2	非导通状态
□<>	ⓢ1 ≠ ⓢ2		□<>	ⓢ1 =ⓢ2	
□<=	ⓢ1 ≤ⓢ2		□<=	ⓢ1 >ⓢ2	
□<	ⓢ1 <ⓢ2		□<	ⓢ1 ≥ⓢ2	
□>=	ⓢ1 ≥ⓢ2		□>=	ⓢ1 <ⓢ2	
□>	ⓢ1 >ⓢ2		□>	ⓢ1 ≤ⓢ2	

(3) ⑤1, ⑤2 中指定了 16 进制数的常数的情况下，如果指定了使最高位 (b15) 变为 1 的数值 (8 ～ F)，将被视为 BIN 值的负数进行比较运算。

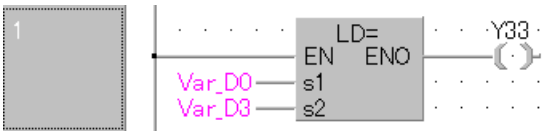
出 错

不存在 =、<>、<=、<、>=、> 指令相关的运算出错。

程序示例

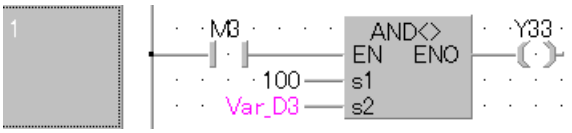
(1) 以下为将 Var_D0 的数据与 Var_D3 的数据进行比较，Var_D0 的数据与 Var_D3 的数据一致时，将 Y33 置为 ON 的程序。

[结构体梯形图]



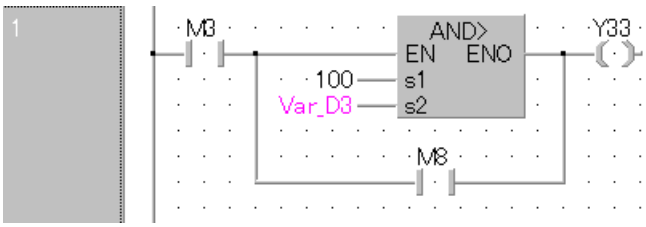
(2) 以下为 M3 为 ON 且将 BIN 值的 100 与 Var_D3 的数据进行比较，Var_D3 的数据大于 100 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



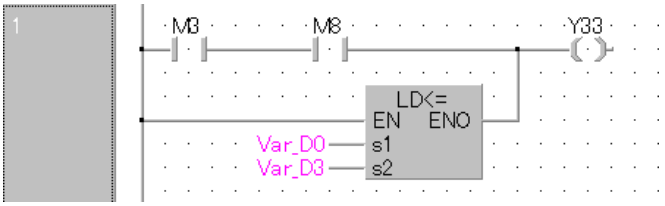
(3) 以下为 M3 为 ON 且将 BIN 值的 100 与 Var_D3 的数据进行比较，Var_D3 的数据小于 100 时，或 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 Var_D0 与 Var_D3 的数据进行比较，Var_D3 的数据大于 Var_D0 的数据时，或 M3 及 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



6.1.2 BIN32 位数据比较

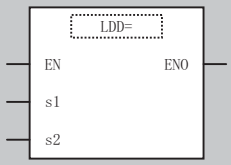
□D=, □D<>, □D<=, □D<, □D>=, □D>

Basic | High performance | Universal | L CPU

LDD (=)
ANDD (<>)
ORD (<=)
(<)
(>=)
(>)

□D=	: =
□D<>	: ≠
□D<=	: ≤
□D<	: <
□D>=	: ≥
□D>	: >

结构体梯形图



ST

不支持。

[ ] 中放入下述指令。

LDD=	ANDD=	ORD=
LDD<>	ANDD<>	ORD<>
LDD<=	ANDD<=	ORD<=
LDD<	ANDD<	ORD<
LDD>=	ANDD>=	ORD>=
LDD>	ANDD>	ORD>

输入自变量， EN: 执行条件 : 位
s1, s2: 比较数据 : ANY32
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○					-
ⓢ2				○					-

功 能

(1) 将ⓢ1 中指定的软元件的 BIN32 位数据与ⓢ2 中指定的软元件的 BIN32 位数据通过常开触点进行比较运算。

(2) 各指令的比较运算结果如下所示。

□内指令符号	条件	比较运算结果	□内指令符号	条件	比较运算结果
□D=	ⓢ1 =ⓢ2	导通状态	□D=	ⓢ1 ≠ⓢ2	非导通状态
□D<>	ⓢ1 ≠ⓢ2		□D<>	ⓢ1 =ⓢ2	
□D<=	ⓢ1 ≤ⓢ2		□D<=	ⓢ1 >ⓢ2	
□D<	ⓢ1 <ⓢ2		□D<	ⓢ1 ≥ⓢ2	
□D>=	ⓢ1 ≥ⓢ2		□D>=	ⓢ1 <ⓢ2	
□D>	ⓢ1 >ⓢ2		□D>	ⓢ1 ≤ⓢ2	

(3) ⑤1, ⑤2 中指定了 16 进制数的常数的情况下，如果指定了使最高位 (b31) 变为 1 的数值 (8 ～ F)，将被视为 BIN 值的负数进行比较运算。

(4) 用于比较的数据应通过 32 位指令 (DMOV 指令等) 进行指定。
通过 16 位指令 (MOV 指令等) 进行指定的情况下，大小比较将无法正常进行。

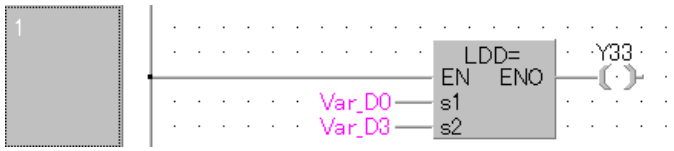
出 错

不存在 D=、D<>、D<=、D<、D>=、D> 指令相关的运算出错。

程序示例

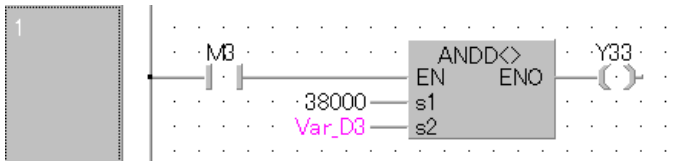
(1) 以下为将 Var_D0 的数据与 Var_D3 的数据进行比较，Var_D0 的数据与 Var_D3 的数据一致时，将 Y33 置为 ON 的程序。

[结构体梯形图]



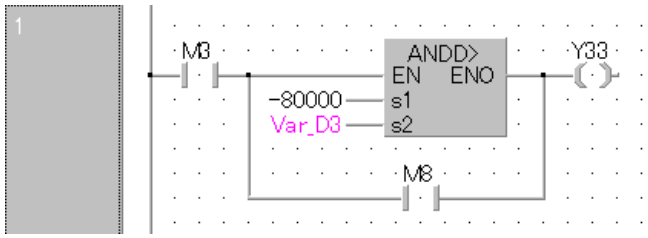
(2) 以下为 M3 为 ON 且将 BIN 值的 38000 与 Var_D3 的数据进行比较，Var_D3 的数据大于 38000 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



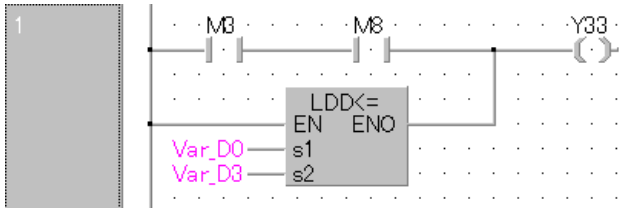
(3) 以下为 M3 为 ON 且将 BIN 值的 -80000 与 Var_D3 的数据进行比较，Var_D3 的数据小于 -80000 时，或 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 Var_D0 与 Var_D3 的数据进行比较 , Var_D3 的数据大于 Var_D0 的数据时，或 M3 及 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



6.1.3 浮动小数点数据比较（单精度）

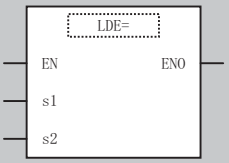
□E=, □E<>, □E<=, □E<, □E>=, □E>

Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

LD	(E=)	□E=	: =
AND	(E<>)	□E<>	: ≠
OR	(E<=)	□E<=	: ≤
	(E<)	□E<	: <
	(E>=)	□E>=	: ≥
	(E>)	□E>	: >

结构体梯形图



ST

不支持。

□ 中放入下述指令。

LDE=	ANDE=	ORE=
LDE<>	ANDE<>	ORE<>
LDE<=	ANDE<=	ORE<=
LDE<	ANDE<	ORE<
LDE>=	ANDE>=	ORE>=
LDE>	ANDE>	ORE>

输入自变量， EN: 执行条件 ：位

s1, s2: 比较数据 ：单精度实数

输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ1	-	○		-	○		_	○	-
ⓢ2	-	○		-	○		_	○	-

功 能

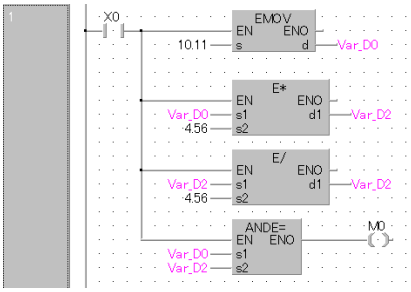
(1) 将ⓢ1中指定的软元件的浮动小数点型数据与ⓢ2中指定的软元件的浮动小数点型数据通过常开触点进行比较运算。

(2) 各指令的比较运算结果如下所示。

□内指令符号	条件	比较运算结果	□内指令符号	条件	比较运算结果
□E=	ⓢ1 =ⓢ2	导通状态	□E=	ⓢ1 ≠ ⓢ2	非导通状态
□E<>	ⓢ1 ≠ ⓢ2		□E<>	ⓢ1 =ⓢ2	
□E<=	ⓢ1 ≤ⓢ2		□E<=	ⓢ1 >ⓢ2	
□E<	ⓢ1 <ⓢ2		□E<	ⓢ1 ≥ⓢ2	
□E>=	ⓢ1 ≥ⓢ2		□E>=	ⓢ1 <ⓢ2	
□E>	ⓢ1 >ⓢ2		□E>	ⓢ1 ≤ⓢ2	

要 点

使用了□ E= 指令的情况下，有可能由于误差而导致无法相等，应加以注意。



出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时）
  （出错代码： 4100）

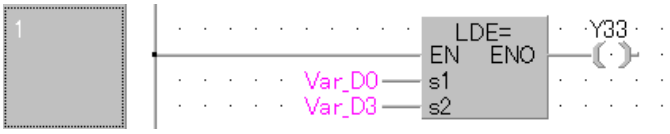
*1 ： 有的 CPU 模块即使被指定为 -0 也不会变为运算出错状态。
详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  $0,\ 2^{-126} \leqslant |$ 指定软元件的内容 $| < 2^{128}$ （出错代码： 4140）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码： 4140）

程序示例

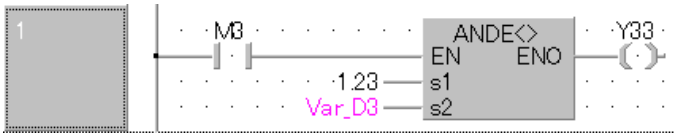
(1) 以下为将 Var_D0 的数据与 Var_D3 的数据进行比较，Var_D0 的数据与 Var_D3 的数据一致时，将 Y33 置为 ON 的程序。

[结构体梯形图]



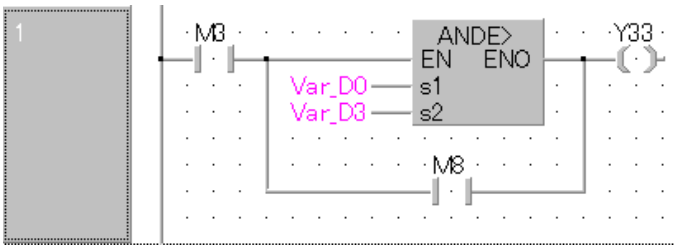
(2) 以下为 M3 为 ON 且将浮动小数点型实数的 1.23 与 Var_D3 的数据进行比较，Var_D3 的数据为除 1.23 以外时，将 Y33 置为 ON 的程序。

[结构体梯形图]



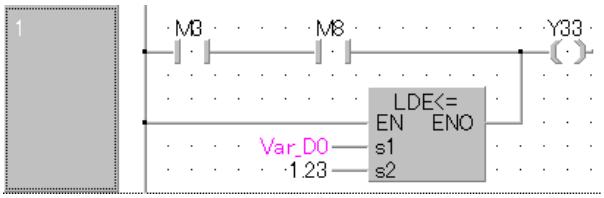
(3) 以下为 M3 为 ON 且将 Var_D0 的数据与 Var_D3 的数据进行比较，Var_D3 的数据小于 Var_D1 的数据时，或 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 Var_D0 的数据与浮动小数点型实数的 1.23 进行比较，1.23 大于 Var_D0 的数据时，或 M3 及 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



6.1.4 浮动小数点数据比较（双精度）

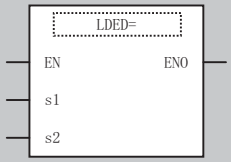
□ED=, □ED<> , □ED<= , □ED<, □ED>=, □ED>

Basic (×) High performance (×) Universal L CPU

LD	(ED=)
AND	(ED<>)
OR	(ED<=)
	(ED<)
	(ED>=)
	(ED>)

□ED=	: =
□ED<>	: ≠
□ED<=	: ≤
□ED<	: <
□ED>=	: ≥
□ED>	: >

结构体梯形图



ST

不支持。

[ ] 中放入下述指令。

LDED=	ANDED=	ORED=
LDED<>	ANDED<>	ORED<>
LDED<=	ANDED<=	ORED<=
LDED<	ANDED<	ORED<
LDED>=	ANDED>=	ORED>=
LDED>	ANDED>	ORED>

输入自变量， EN: 执行条件 ：位

s1, s2: 比较数据 ：双精度实数

输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ1	-	○		-				○	-
ⓢ2	-	○		-				○	-

功 能

(1) 将ⓢ1 中指定的 64 位浮动小数点型实数数据与ⓢ2 中指定的 64 位浮动小数点型实数数据通过常开触点进行比较运算。

(2) 各指令的比较运算结果如下所示。

□内指令符号	条件	比较运算结果	□内指令符号	条件	比较运算结果
□ED=	ⓢ1 =ⓢ2	导通状态	□ED=	ⓢ1 ≠ⓢ2	非导通状态
□ED<>	ⓢ1 ≠ⓢ2		□ED<>	ⓢ1 =ⓢ2	
□ED<=	ⓢ1 ≤ⓢ2		□ED<=	ⓢ1 >ⓢ2	
□ED<	ⓢ1 <ⓢ2		□ED<	ⓢ1 ≥ⓢ2	
□ED>=	ⓢ1 ≥ⓢ2		□ED>=	ⓢ1 <ⓢ2	
□ED>	ⓢ1 >ⓢ2		□ED>	ⓢ1 ≤ⓢ2	

出 错

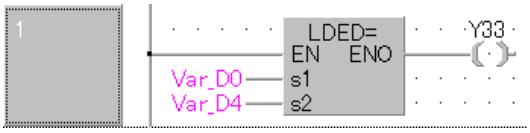
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqslant |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）

程序示例

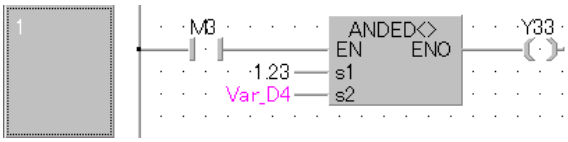
(1) 以下为将 Var_D0 的数据与 Var_D4 的数据进行比较，Var_D0 的数据与 Var_D4 的数据一致时，将 Y33 置为 ON 的程序。

[结构体梯形图]



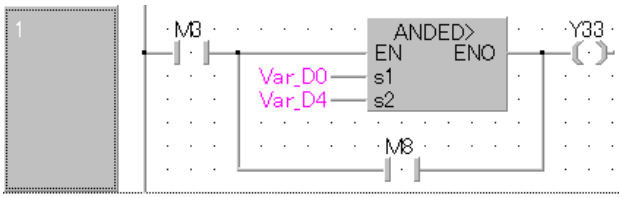
(2) 以下为 M3 为 ON 且将浮动小数点型实数的 1.23 与 Var_D4 的数据进行比较，Var_D4 的数据为 1.23 以外时，将 Y33 置为 ON 的程序。

[结构体梯形图]



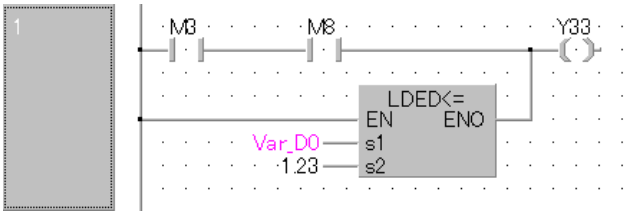
(3) 以下为 M3 为 ON 且将 Var_D0 的数据与 Var_D4 的数据进行比较，Var_D4 的数据小于 Var_D0 时，或 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 Var_D0 的数据与浮动小数点型实数的 1.23 进行比较，1.23 大于 Var_D0 的数据时，或 M3 及 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]

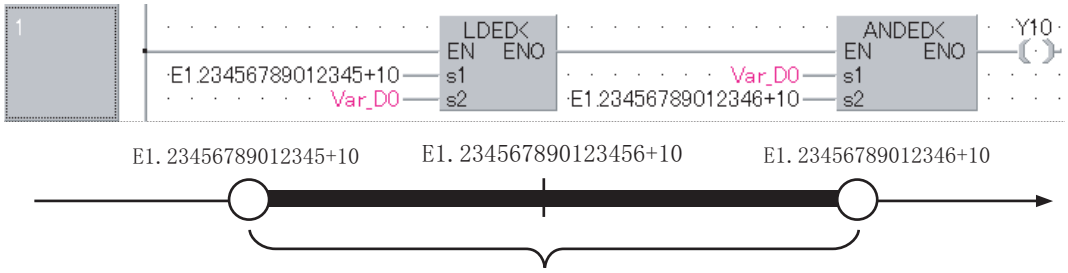


注意事项

由于通过编程工具可输入的实数的位数最多为 15 位，因此在本项中所示的指令中不能与有效位数 16 位以上的实数进行比较。

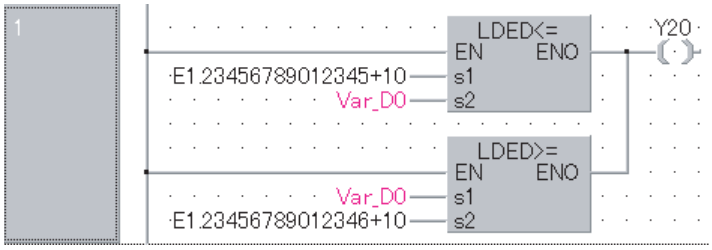
通过本项的指令对有效位数为 16 位以上的实数进行一致与否的判断的情况下，需要通过与要比较的实数前后的近似值进行大小比较来加以判断。

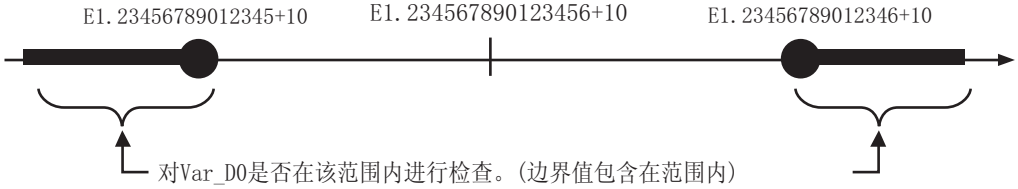
例 对 E1.234567890123456+10（有效位数 16 位）与双精度浮动小数点数据进行一致判断的情况下。



对Var_D0是否在该范围内进行检查。(边界值不包含在范围内)

例 对 E1.234567890123456+10（有效位数 16 位）与双精度浮动小数点数据的不一致进行判断的情况下。





对Var_D0是否在该范围内进行检查。(边界值包含在范围内)

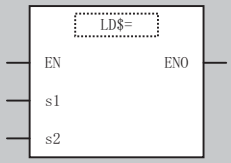
6.1.5 字符串数据比较

□$=, □$<>, □$<=, □$<, □$>=, □$>

Basic | High performance | Universal | L CPU

LD$	(=)	□$= : =
AND$	(<>)	□$<> : ≠
OR$	(<=)	□$<= : ≤
	(<)	□$< : <
	(>=)	□$>= : ≥
	(>)	□$> : >

结构体梯形图



ST

不支持。

□ 中放入下述指令。

LD$=	AND$=	OR$=
LD$<>	AND$<>	OR$<>
LD$<=	AND$<=	OR$<=
LD$<	AND$<	OR$<
LD$>=	AND$>=	OR$>=
LD$>	AND$>	OR$>

输入自变量，	EN:	执行条件	: 位
	s1, s2:	比较数据	: 字符串
输出自变量，	ENO:	执行结果	: 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ1	-	○		-				○	-
ⓢ2	-	○		-				○	-

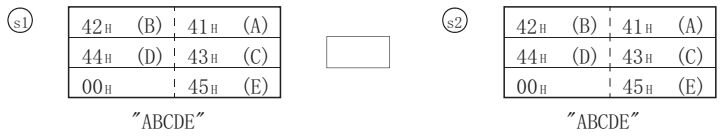
功 能

(1) 将ⓢ1 中指定的字符串数据与ⓢ2 中指定的字符串数据通过常开触点进行比较运算。

(2) 进行比较运算时，将字符串的 ASCII 码从字符串的最开始处逐个字符进行比较。

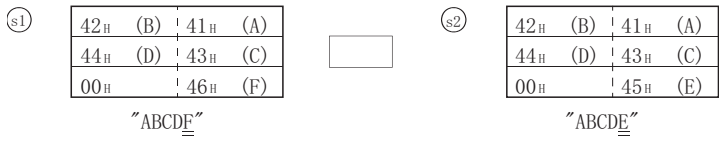
(3) ⓢ1, ⓢ2 的字符串到 00H 为止将作为比较对象。

(a) 所有的字符串一致的情况下，比较结果变为一致。



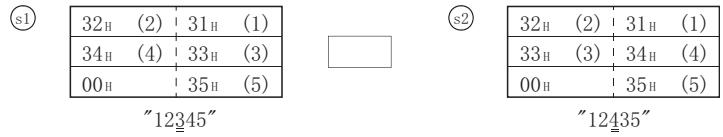
□内指令符号	比较运算结果	□内指令符号	比较运算结果
□$=	导通状态	□$<	非导通状态
□$<>	非导通状态	□$>=	导通状态
□$<=	导通状态	□$>	非导通状态

(b) 不同的字符串的情况下，字符代码较大的字符串为大。



□内指令符号	比较运算结果	□内指令符号	比较运算结果
□$=	非导通状态	□$<	非导通状态
□$<>	导通状态	□$>=	导通状态
□$<=	非导通状态	□$>	导通状态

(c) 不同字符串的情况下，以最初的字符代码的大小来确定字符串的大小。



□内指令符号	比较运算结果	□内指令符号	比较运算结果
□$=	非导通状态	□$<	导通状态
□$<>	导通状态	□$>=	非导通状态
□$<=	导通状态	□$>	非导通状态

(d) ⓢ1 与ⓢ2 中的字符串数据的长度不同的情况下，较长的字符串数据为大。

ⓢ1

32H	(2)	31H	(1)
34H	(4)	33H	(3)
36H	(6)	35H	(5)
00H		37H	(7)

"1234567"

ⓢ2

32H	(2)	31H	(1)
34H	(4)	33H	(3)
36H	(6)	35H	(5)
00H		00H	

"123456"

□ 内指令符号	比较运算结果	□ 内指令符号	比较运算结果
□$=	非导通状态	□$<	非导通状态
□$<>	导通状态	□$>=	导通状态
□$<=	非导通状态	□$>	导通状态

出 错

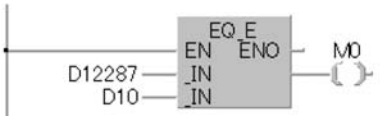
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1， ⓢ2 中指定的软元件编号以后，相应软元件的范围中不存在 00H 时。
  （出错代码： 4101）
- ⓢ1， ⓢ2 字符串超过了 16383 个字符时。 （出错代码： 4101）

要 点

在字符串数据比较运算指令中，在进行字符串的比较的同时进行软元件范围检查。因此即使相应软元件的范围中不存在 00H 时，如果在软元件范围内检测出字符的不一致，将不变为运算出错状态，对比较运算结果进行输出。

例



ⓢ1的数据

D12287	"B"	"A"
W0	00H	"C"

ⓢ2的数据

D10	"Z"	"A"
D11	00H	"C"

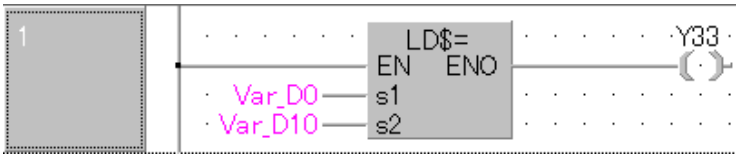
ⓢ1 与ⓢ2 数据处于上述情况时，由于ⓢ1 的第 2 个字符与ⓢ2 的不相同，因此变为ⓢ1 ≠ ⓢ2，运算结果变为非导通。

虽然此时ⓢ1 软元件的范围中不存在 00H，但由于检测出不一致的软元件为 D12287（在软元件的范围内），因此不变为运算出错状态。

程序示例

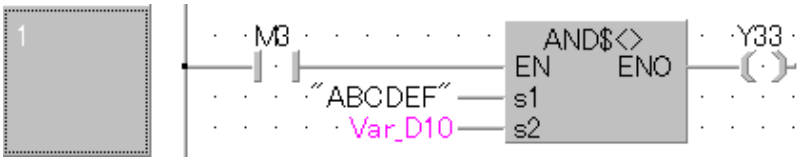
(1) 以下为将 Var_D0 以后存储的字符串与 Var_D10 以后存储的字符串进行比较，Var_D0 以后存储的字符串与 Var_D10 以后存储的字符串一致时，将 Y33 置为 ON 的程序。

[结构体梯形图]



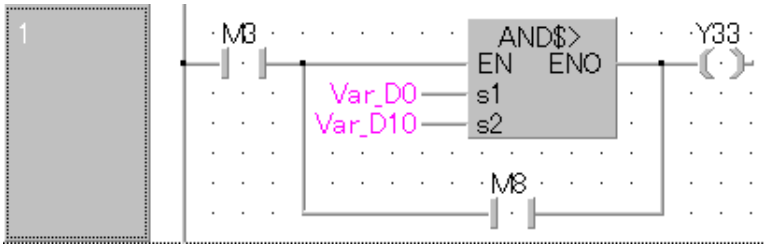
(2) 以下为 M3 为 ON 且将字符串的 “ABCDEF” 与 Var_D10 以后存储的字符串进行比较，Var_D10 以后存储的字符串为除 “ABCDEF” 以外时，将 Y33 置为 ON 的程序。

[结构体梯形图]



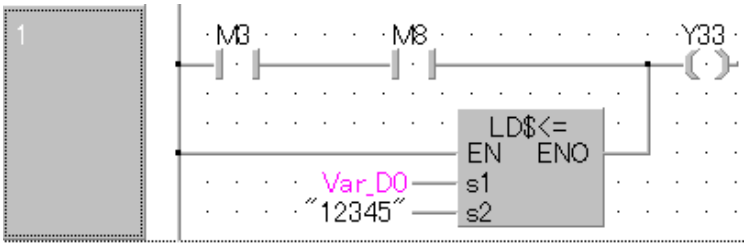
(3) 以下为 M3 为 ON 且将 Var_D0 以后存储的字符串与 Var_D10 以后存储的字符串进行比较，Var_D10 以后存储的字符串小于 Var_D0 以后存储的字符串时，或 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 Var_D0 以后存储的字符串与 “12345” 的字符串进行比较，“12345” 大于 Var_D0 以后存储的字符串时，或 M3 及 M8 为 ON 时，将 Y33 置为 ON 的程序。

[结构体梯形图]



6.1.6 BIN16 位块数据比较

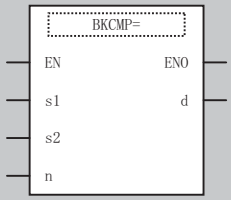
BKCMP=, BKCMP<>, BKCMP<=,
BKCMP<, BKCMP>=, BKCMP>

Basic | High performance | Universal | L CPU

BKCMP (=) (P)
(<>)
(<=)
(<)
(>=)
(>)

BKCMP= : =
BKCMP<> : ≠
BKCMP<= : ≤
BKCMP< : <
BKCMP>= : ≥
BKCMP> : >
P: 执行条件

结构体梯形图



ST

不支持。

中放入下述指令。

BKCMP=	BKCMP=P
BKCMP<>	BKCMP<>P
BKCMP<=	BKCMP<=P
BKCMP<	BKCMP<P
BKCMP>=	BKCMP>=P
BKCMP>	BKCMP>P

输入自变量，	EN:	执行条件	: 位
	s1	被比较的数据或存储被比较数据的软元件的起始编号	: ANY16
	s2:	存储比较数据的软元件的起始编号	: ANY16
	n	要比较的数据数	: ANY16
输出自变量，	ENO:	执行结果	: 位
	d:	存储运算结果的软元件的起始编号	: 位

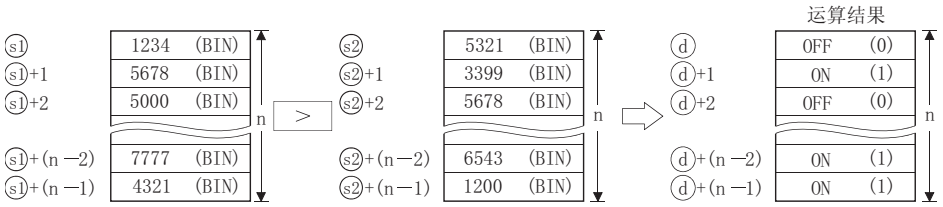
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1	-	○		-				○	-
⑤2	-	○		-				-	-
n	○	○		○				○	-
ⓓ	○	○		-				-	-

功 能

(1) 将⑤1 中指定的软元件编号算起 n 点的 BIN16 位数据与⑤2 中指定的软元件算起 n 点的 BIN16 位数据进行比较，将运算结果存储到ⓓ 中指定的软元件以后。

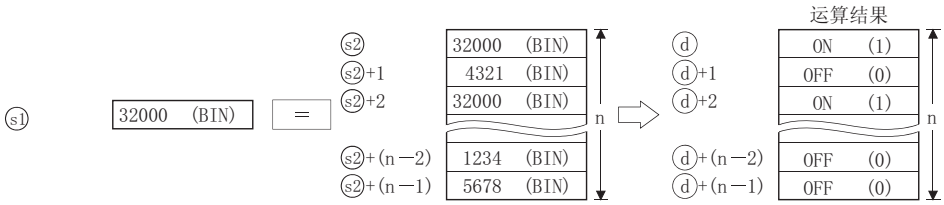
(a) 比较条件成立时，ⓓ 的相应软元件将变为 ON。

(b) 比较条件不不成立时，ⓓ 的相应软元件将变为 OFF。



(2) 比较运算是以 16 位为单位进行。

(3) ⓢ1 中可以指定 -32768 ～ 32767(BIN16 位) 的常数。



(4) 各指令的比较运算结果如下所示。

指令符号	条件	比较运算结果	指令符号	条件	比较运算结果
BKCMP=	ⓢ1 =ⓢ2	ON(1)	BKCMP=	ⓢ1 ≠ ⓢ2	OFF(0)
BKCMP<>	ⓢ1 ≠ ⓢ2		BKCMP<>	ⓢ1 =ⓢ2	
BKCMP<=	ⓢ1 ≤ⓢ2		BKCMP<=	ⓢ1 >ⓢ2	
BKCMP<	ⓢ1 <ⓢ2		BKCMP<	ⓢ1 ≥ⓢ2	
BKCMP>=	ⓢ1 ≥ⓢ2		BKCMP>=	ⓢ1 <ⓢ2	
BKCMP>	ⓢ1 >ⓢ2		BKCMP>	ⓢ1 ≤ⓢ2	

(5) ⓓ 算起 n 点中存储的比较结果全部为 ON(1) 时，SM704(块比较信号) 将变为 ON。

出 错

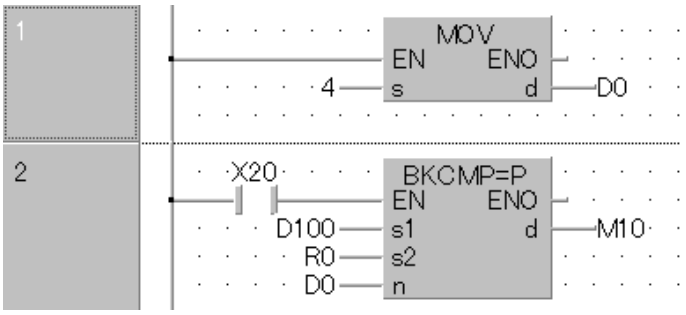
在以下情况下将发生运算出错，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- ⓢ1， ⓢ2， ⓓ 中指定的软元件算起的 n 点超出了指定软元件的范围时。 (出错代码： 4101)
- ⓢ1 算起 n 点的软元件范围与ⓓ 算起 n 点的软元件范围重叠时。 (出错代码： 4101)
- ⓢ2 算起 n 点的软元件范围与ⓓ 算起 n 点的软元件范围重叠时。 (出错代码： 4101)

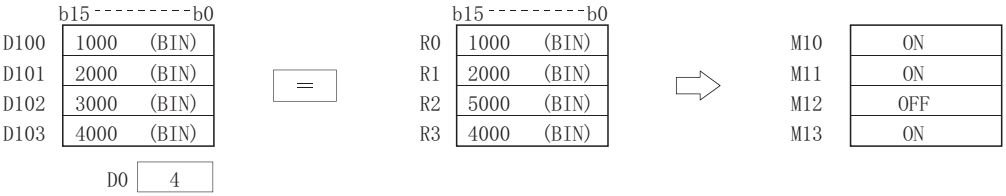
程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D103 中存储的值的数据与 R0 ～ R3 中存储的值的数据进行比较运算，将其结果存储到 M10 以后的程序。

[结构体梯形图]

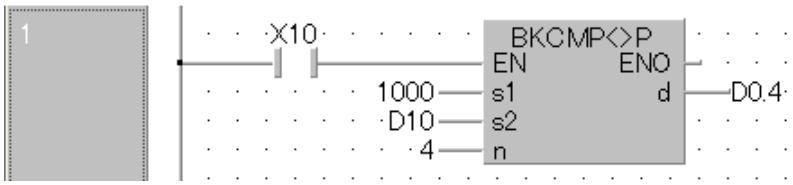


[动作]

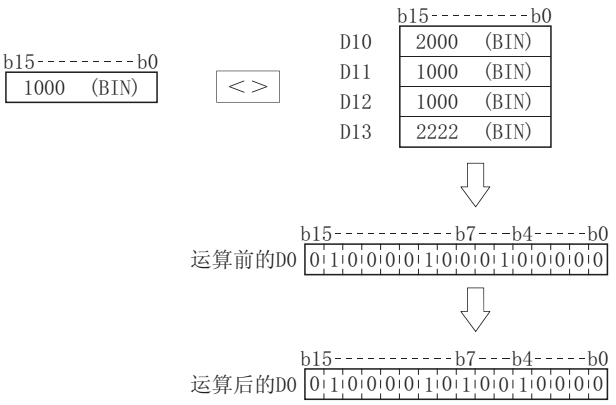


(2) 以下为 X1C 变为 ON 时，将常数 K1000 与 D10 ～ D13 中存储的值的数据进行比较运算，将其结果存储到 D0 的 b4 ～ b7 中的程序。

[结构体梯形图]



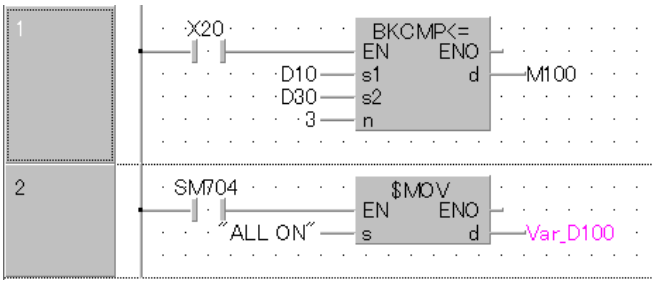
[动作]



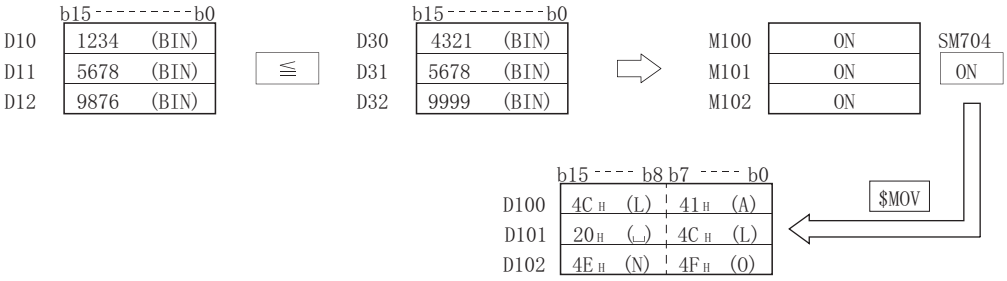
(3) 以下为 X20 变为 ON 时，将 D10 ～ D12 的数据与 D30 ～ D32 的数据进行比较，将其结果存储到 M100 以后的程序。

M100 以后的软元件全部变为 1(ON) 的情况下，将字符串 “ALL ON”传送至 Var_D100 以后。

[结构体梯形图]



[动作]



6.1.7 BIN32 位块数据比较

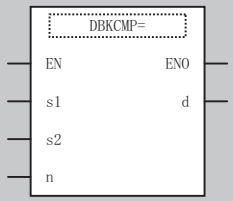
DBKCMP=, DBKCMP<>, DBKCMP<=,
BDKCMP<, DBKCMP>=, DBKCMP>

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

DBKCMP (=) (P)
(<>)
(<=)
(<)
(>=)
(>)

DBKCMP=	: =
DBKCMP<>	: ≠
DBKCMP<=	: ≤
DBKCMP<	: <
DBKCMP>=	: ≥
DBKCMP>	: >
P：执行条件	

结构体梯形图



ST

不支持。

[ ] 中放入下述指令。

DBKCMP=	DBKCMP=P
DBKCMP<>	DBKCMP<>P
DBKCMP<=	DBKCMP<=P
DBKCMP<	DBKCMP<P
DBKCMP>=	DBKCMP>=P
DBKCMP>	DBKCMP>P

输入自变量，	EN:	执行条件	：位
	s1:	被比较的数据或存储被比较数据的软元件的起始编号	: ANY32
	s2:	存储比较数据的软元件的起始编号	: ANY32
	n:	要比较的数据数	: ANY16
输出自变量，	ENO:	执行结果	：位
	d:	存储运算结果的软元件的起始编号	：位

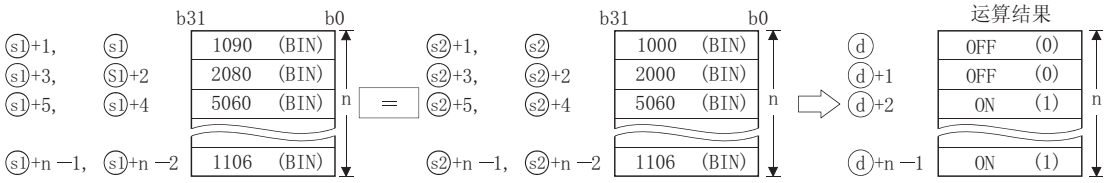
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1	-	○	○	-				○	-
⑤2	-	○	○	-				-	-
n	-	○	○	○				○	-
ⓓ	○	-	○	-				-	-

功 能

(1) 将⑤1 中指定的软元件编号 n 点的 BIN32 位数据与⑤2 中指定的软元件算起 n 点的 BIN32 位数据进行比较，将运算结果存储到ⓓ 中指定的软元件以后。

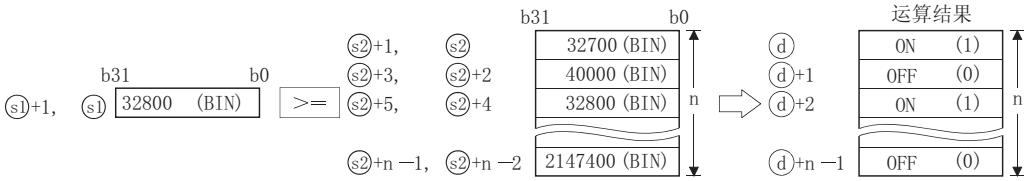
(a) 比较条件成立时，ⓓ 的相应软元件将变为 ON。

(b) 比较条件不成立时，ⓓ 的相应软元件将变为 OFF。



(2) 比较运算是以 32 位为单位进行。

(3) ⓢ1 中可以指定 -2147483648 ～ 2147483647(BIN32 位) 的常数。



(4) ⓓ 的指定应在ⓢ1 算起 n 点的软元件范围及ⓢ2 算起 n 点的软元件范围以外。

(5) 各指令的比较运算结果如下所示。

指令符号	条件	比较运算结果	指令符号	条件	比较运算结果
DBKCMP=	ⓢ1 =ⓢ2	ON(1)	DBKCMP=	ⓢ1 ≠ ⓢ2	OFF(0)
DBKCMP<>	ⓢ1 ≠ ⓢ2		DBKCMP<>	ⓢ1 =ⓢ2	
DBKCMP<=	ⓢ1 ≤ⓢ2		DBKCMP<=	ⓢ1 >ⓢ2	
DBKCMP<	ⓢ1 <ⓢ2		DBKCMP<	ⓢ1 ≥ⓢ2	
DBKCMP>=	ⓢ1 ≥ⓢ2		DBKCMP>=	ⓢ1 <ⓢ2	
DBKCMP>	ⓢ1 >ⓢ2		DBKCMP>	ⓢ1 ≤ⓢ2	

(6) ⓓ 算起 n 点中存储的比较结果全部为 ON(1) 时，或者有的结果为 OFF(0) 时，根据条件特殊继电器的 ON/OFF 如下所示。

No.	编号	比较运算结果全部为 ON(1) 时			比较运算结果中包含有 OFF(0) 时		
		初期执行 / 扫描	中断 (I45 除外)/ 恒定周期执行	中断 (I45)	初期执行 / 扫描	中断 (I45 除外)/ 恒定周期执行	中断 (I45)
1	SM704	ON	ON	ON	OFF	OFF	OFF
2	SM716	ON	—	—	OFF	—	—
3	SM717	—	ON	—	—	OFF	—
4	SM718	—	—	ON	—	—	OFF

待机程序的情况下，根据调用源程序将特殊继电器置为 ON/OFF。

(7) n 中指定的值为 0 的情况下变为无处理。

出 错

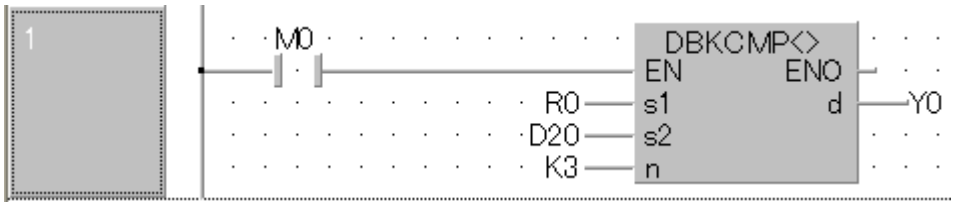
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 中指定了负值时。 (出错代码： 4100)
- ⑤1， ⑤2， ⓓ 中指定的软元件算起的 n 点超出了指定软元件的范围时。 (出错代码： 4101)
- ⑤1 算起 n 点的软元件范围与 ⓓ 算起 n 点的软元件范围重叠时。 (出错代码： 4101)
- ⑤2 算起 n 点的软元件范围与 ⓓ 算起 n 点的软元件范围重叠时。 (出错代码： 4101)

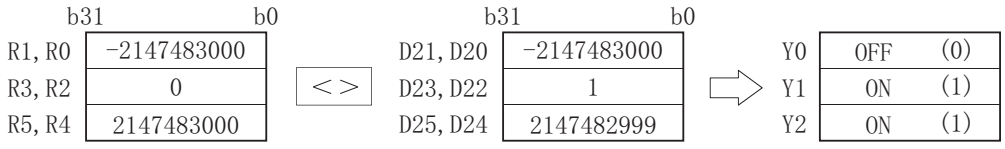
程序示例

(1) 以下为 M0 变为 ON 时，将 R0 ～ R5 中存储的值与 D20 ～ D25 中存储的值进行比较，将运算结果存储到 Y0 ～ Y2 中的程序。

[结构体梯形图]

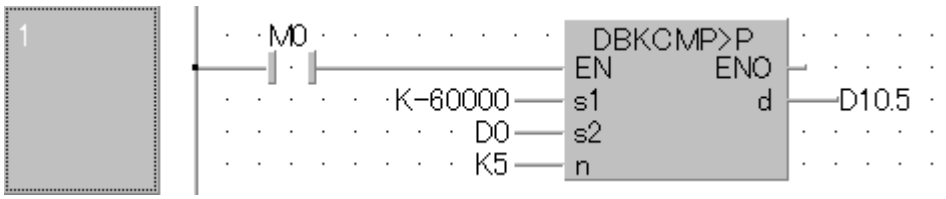


[动作]

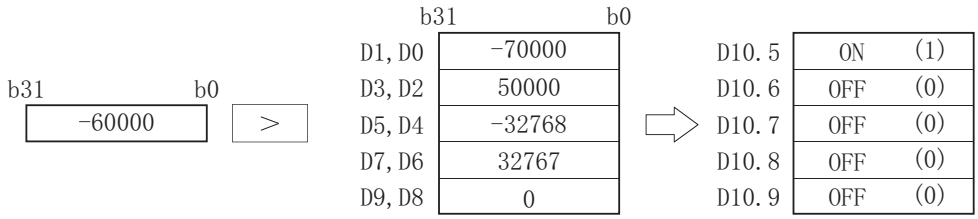


(2) 以下为 M0 变为 ON 时，将常数与 D0 ～ D9 中存储的值进行比较，将运算结果存储到 D10.5 ～ D10.9 中的程序。

[结构体梯形图]

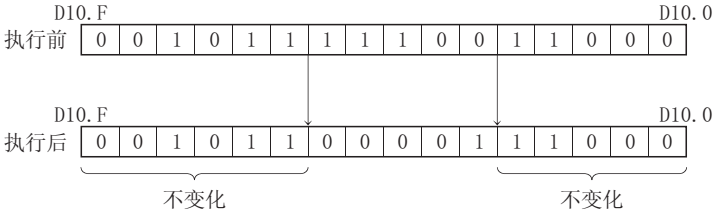


[动作]



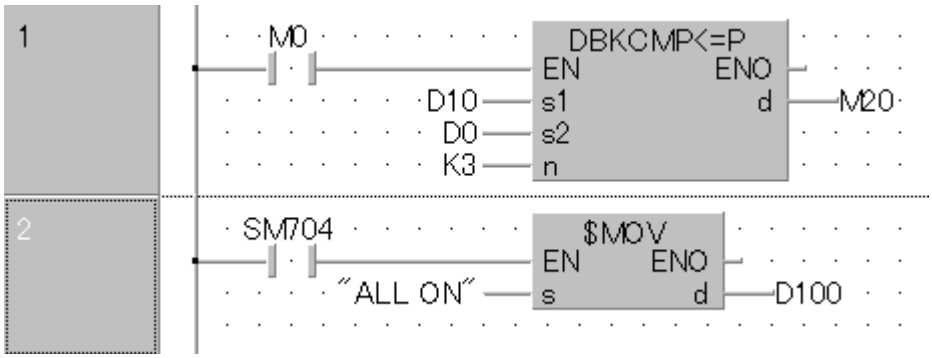
要 点

进行了字软元件的位指定的情况下，除存储运算结果的位指定软元件以外不发生变化。

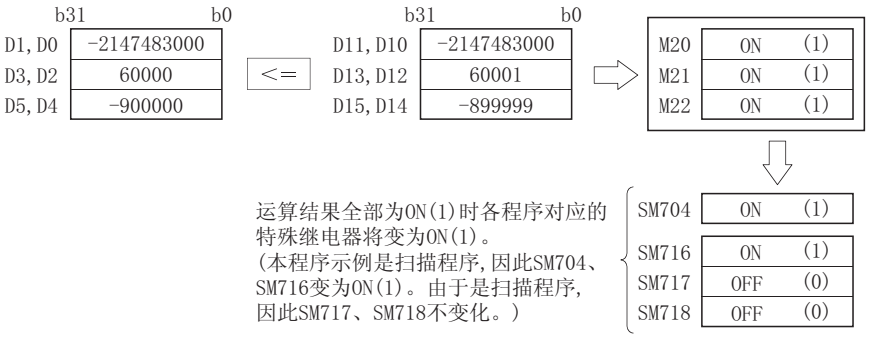


(3) 以下为 M0 变为 ON 时，将 D0 ～ D5 中存储的值与 D10 ～ D15 中存储的值进行比较，将运算结果存储到 M20 ～ M22 中，M20 ～ M22 的软元件全部变为 ON 时，将字符串 “ALL ON”传送至 D100 以后的扫描程序。

[结构体梯形图]



[动作]



6.2 算术运算指令

6.2.1 BIN16 位加减法

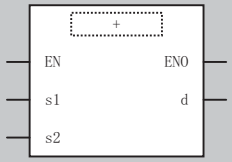
+, −

Basic | High performance | Universal | L CPU

+(P)
−(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

+	+P
−	−P

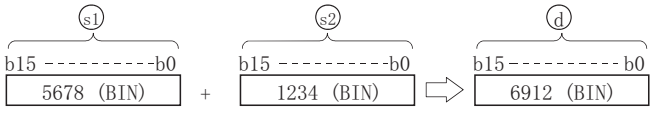
输入自变量，EN：　执行条件　：位
　　s1：　被进行加减法的数据或存储被进行加减法的数据的软元件的起始编号　：ANY16
　　s2：　存储加减法数据的软元件的起始编号　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1		○						○	−
ⓢ2		○						○	−
ⓓ		○						−	−

功 能

+(P)

(1) 将ⓢ1 中指定的 BIN16 位数据与ⓢ2 中指定的 BIN16 位数据进行加法运算，将结果存储到ⓓ 中指定的变量中。



(2) ⓢ1，ⓢ2，ⓓ 中可在 -32768 ～ 32767(BIN16 位) 的范围内指定。

(3) 数据的正负判定是由最高位 (b15) 进行。

• 0..... 正

• 1..... 负

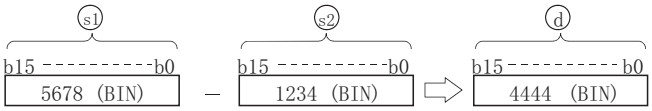
(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• 32767 (7FFFH) +2 (0002H) ⟶ −32767·····b15变为1,因此变为负值。 (8001H)

• −32768 (8000H) +−2 (FFFEH) ⟶ 32766 ·····b15变为0,因此变为正值。 (7FFEH)

−(P)

(1) 将ⓢ1 中指定的 BIN16 位数据与ⓢ2 中指定的 BIN16 位数据进行减法运算，将结果存储到ⓓ 中指定的变量中。



(2) ⓢ1 , ⓢ2 , ⓓ 中可在 -32768 ～ 32767(BIN16 位) 的范围内指定。

(3) 数据的正负判定是由最高位 (b15) 进行。

• 0..... 正

• 1..... 负

(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

• −32768 (8000H) −2 (0002H) ⟶ 32766·····b15变为0,因此变为正值。 (7FFEH)

• 32767 (7FFFH) −2 (FFFEH) ⟶ −32767 ·····b15变为1,因此变为负值。 (8001H)

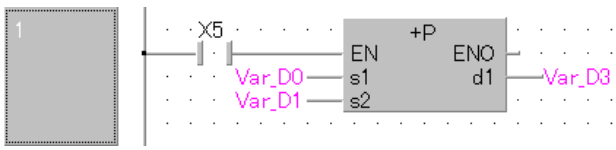
出 错

不存在 +(P)、-(P) 指令相关的运算出错。

程序示例

以下为 X5 变为 ON 时，对 Var_D0 与 Var_D1 的内容进行加法运算后，输出到 Var_D3 中的程序。

[结构体梯形图]



6.2.2 BIN32 位加减法

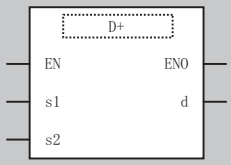
D+, D-

Basic | High performance | Universal | L CPU

D+(P)
D-(P)

P: 执行条件 : ↑

结构体梯形图



ST

不支持。

中放入下述指令。

D+	D+P
D-	D-P

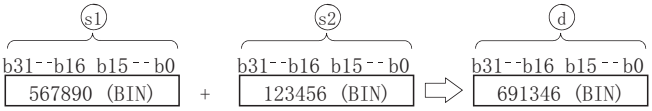
输入自变量， EN: 执行条件 : 位
s1: 被进行加减法的数据 : ANY32
s2: 加减法数据 : ANY32
输出自变量， ENO: 执行结果 : 位
d: 运算结果 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ				○				-	-

功 能

D+(P)

(1) 将ⓢ1 中指定的 BIN32 位数据与ⓢ2 中指定的 BIN32 位数据进行加法运算，将加法结果存储到ⓓ 中指定的变量中。



(2) ⓢ1, ⓢ2, ⓓ 中可在 -2147483648 ～ 2147483647(BIN32 位) 的范围进行指定。

(3) 数据的正负判定是由最高位 (b31) 进行。

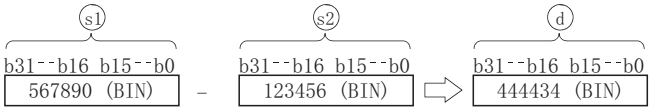
- 0 ..... 正
- 1 ..... 负

(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- 2147483647 (7FFFFFFFH) +2 (00000002H) → −2147483647···b31变为1，因此变为负值。 (80000001H)
- −2147483648 (80000000H) +−2 (FFFFFFFEH) → 2147483646 ····b31变为0，因此变为正值。 (7FFFFFFEH)

D-(P)

(1) 将㉑ 中指定的 BIN32 位数据与㉒ 中指定的 BIN32 位数据进行减法运算，将结果存储到ⓓ 中指定的变量中。



(2) ㉑， ㉒， ⓓ 中可在 -2147483648 ～ 2147483647(BIN32 位) 的范围内进行指定。

(3) 数据的正负判定是由最高位 (b31) 进行。

- 0..... 正
- 1..... 负

(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- −2147483648 (80000000H) −2 (00000002H) → 2147483646 ····b31变为0，因此变为正值。 (7FFFFFFEH)
- 2147483647 (7FFFFFFFH) −−2 (FFFFFFFEH) → −2147483647···b31变为1，因此变为负值。 (80000001H)

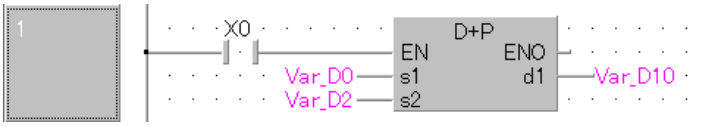
出 错

不存在 D+(P)、D-(P) 指令相关的运算出错。

程序示例

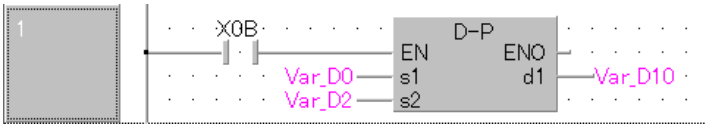
(1) 以下为 X0 变为 ON 时，将 Var_D0 的数据与 Var_D2 的数据进行加法运算，将结果存储到 Var_D10 中的程序。

[结构体梯形图]



(2) 以下为 X0B 变为 ON 时，将 Var_D0 的数据与 Var_D2 的数据进行减法运算，将结果存储到 Var_D10 中的程序。

[结构体梯形图]



6.2.3 BIN16 位乘除法

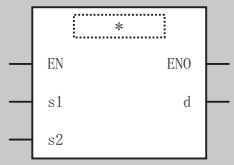
*, /

Basic | High performance | Universal | L CPU

*(P)
/(P)

P: 执行条件 : ↑

结构体梯形图



ST

不支持。

中放入下述指令。

*	*P
/	/P

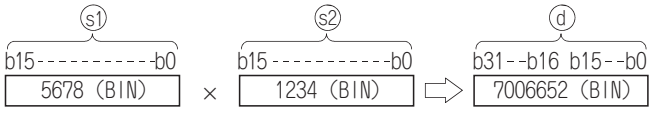
输入自变量， EN: 执行条件 : 位
s1: 被进行乘除法的数据或存储被进行乘除的数据的软元件的起始编号 : ANY16
s2: 乘除法数据或存储乘除法数据的软元件的起始编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 运算结果 : ANY32, ANY16 的数组 (0..1)

设置数据	内部软元件		R ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ				○				-	-

功 能

*(P)

(1) 将ⓢ1 中指定的 BIN16 位数据与ⓢ2 中指定的 BIN16 位数据进行乘法运算，将乘法运算的结果存储到ⓓ 中指定的变量中。



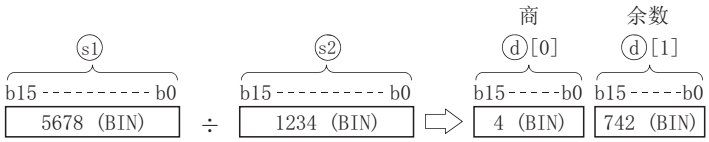
(2) ⓢ1, ⓢ2 中可在 -32768 ～ 32767(BIN16 位) 的范围内进行指定。

(3) ⓢ1, ⓢ2, ⓓ 的数据中正负判定是由最高位 (ⓢ1, ⓢ2 为 b15，ⓓ 为 b31) 进行的。

- 0 ..... 正
- 1 ..... 负

/(P)

(1) 将㉑ 中指定的 BIN16 位数据与㉒ 中指定的 BIN16 位数据进行除法运算，将除法运算结果存储到ⓓ 中指定的变量中。



(2) 对于除算结果，使用 32 位存储商及余数。

商 ..... 被存储到ⓓ [0](16 位) 中。

余数 ..... 被存储到ⓓ [1](16 位) 中。

(3) ㉑ , ㉒ 中可在 -32768 ～ 32767(BIN16 位) 的范围内进行指定。

(4) ㉑ , ㉒ , ⓓ 的数据中正负判定是由最高位 (b15) 进行。
(商及余数均带符号。)

- 0..... 正
- 1..... 负

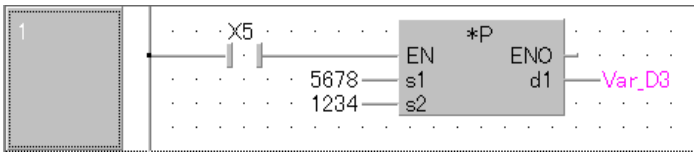
出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

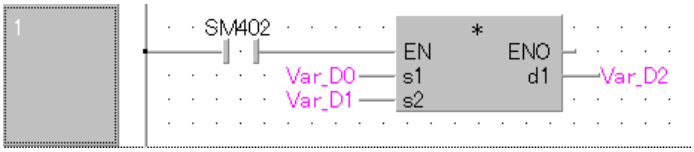
- 除数㉒ 为 0 时。 (出错代码： 4100)

程序示例

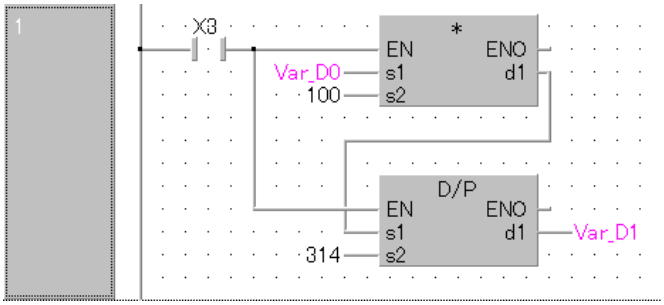
(1) 以下为 X5 变为 ON 时，将 BIN 的 5678 与 1234 的乘法运算结果存储到 Var_D3 中的程序。
[结构体梯形图]



(2) 以下为将 Var_D0 的 BIN 数据与 Var_D1 的 BIN 数据的乘法运算结果存储到 Var_D2 中的程序。
[结构体梯形图]



(3) 以下为 X3 变为 ON 时，将 Var_D0 的数据用 3.14 相除，将结果存储到 Var_D1 中的程序。
[结构体梯形图]



6.2.4 BIN32 位乘除法

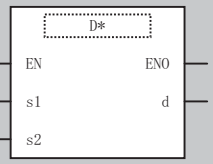
D*, D/

Basic | High performance | Universal | L CPU

D*(P)
D/(P)

(P: 执行条件 : ↵)

结构体梯形图



ST

不支持。

[ ] 中放入下述指令。

D*	D*P
D/	D/P

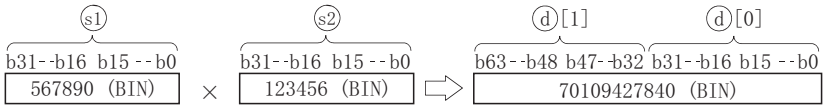
输入自变量， EN: 执行条件 : 位
s1: 被进行乘除法的数据 : ANY32
s2: 乘除法数据 : ANY32
输出自变量， ENO: 执行结果 : 位
d: 运算结果 : ANY32 的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	○			○					-
ⓢ2	○			○					-
ⓓ	○			-					-

功 能

D*(P)

(1) 将ⓢ1 中指定的 BIN32 位数据与ⓢ2 中指定的 BIN32 位数据进行乘法运算，将结果存储到ⓓ 中指定的变量中。



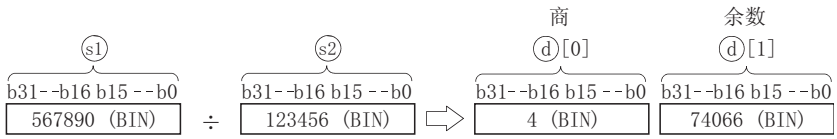
(2) ⓢ1, ⓢ2 中可在 -2147483648 ～ 2147483647(BIN32 位) 的范围内进行指定。

(3) ⓢ1, ⓢ2, ⓓ 的数据中正负判定是由最高位 (ⓢ1, ⓢ2 为 b31, ⓓ 为 b63) 进行。

- 0 ..... 正
- 1 ..... 负

D/(P)

(1) 将ⓢ1 中指定的 BIN 数据与ⓢ2 中指定的 BIN 数据进行除法运算，将结果存储到ⓓ 中指定的变量中。



(2) 对于除法运算结果，使用 64 位存储商余数。

商 ..... 被存储到ⓓ [0](32 位) 中。

余数 .... 被存储到ⓓ [1](32 位) 中。

(3) ⓢ1 , ⓢ2 中可在 -2147483648 ～ 2147483647(BIN32 位) 的范围内进行指定。

(4) ⓢ1 , ⓢ2 , ⓓ 的数据中正负判定是由最高位 (b31) 进行。

(商及余数均带符号。)

- 0..... 正
- 1..... 负

出 错

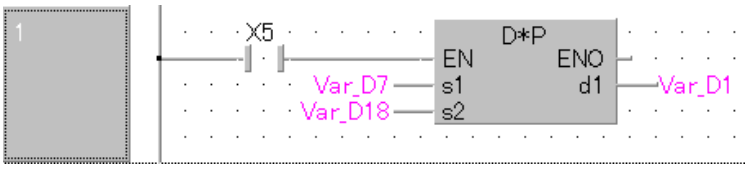
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 除数ⓢ2 为 0 时。 (出错代码： 4100)

程序示例

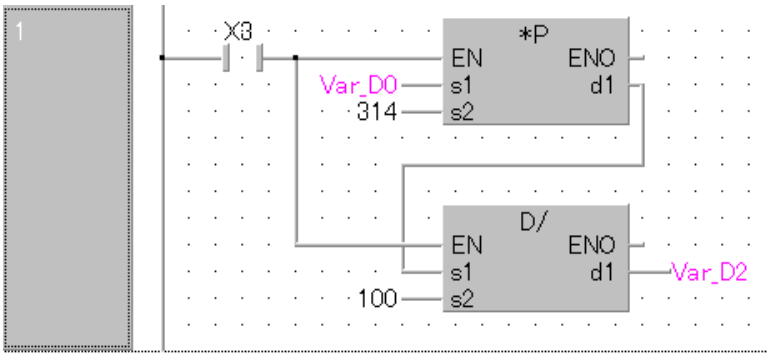
(1) 以下为 X5 变为 ON 时，将 Var_D7 的 BIN 数据与 Var_D18 的 BIN 数据的乘法运算结果存储到 Var_D1 中的程序。

[结构体梯形图]



(2) 以下为 X3 变为 ON 时，将 Var_D0 的数据用 3.14 相乘并将结果存储到 Var_D2 中的程序。

[结构体梯形图]



6.2.5 BCD4 位加减法

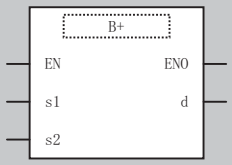
B+, B-

Basic | High performance | Universal | L CPU

B+(P)
B-(P)

P: 执行条件 : ↓

结构体梯形图



ST

不支持。

中放入下述指令。

B+ B+P
B- B-P

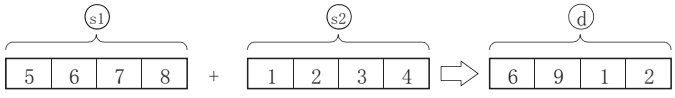
输入自变量， EN: 执行条件 : 位
s1: 被进行加减法的数据或存储被进行加减法的数据的软元件的起始编号 : ANY16
s2: 存储加减法数据的软元件的起始编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储运算结果的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ				○				-	-

功 能

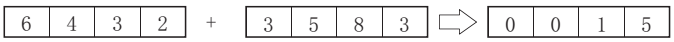
B+(P)

(1) 将ⓢ1 中指定的 BCD4 位数据与ⓢ2 中指定的 BCD4 位数据进行加法运算，将加法结果存储到ⓓ 中指定的变量中。



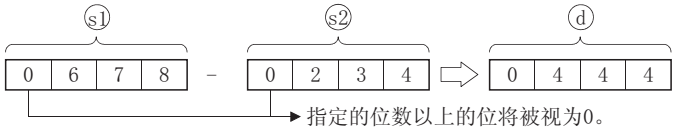
(2) ⓢ1, ⓢ2, ⓓ 中可在 0 ～ 9999(BCD4 位) 的范围内进行指定。

(3) 加法运算结果超过了 9999 的情况下，进位将被忽略。
在这种情况下，进位标志不变为 ON。



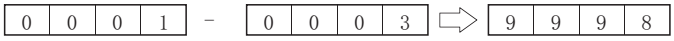
B-(P)

(1) 将㉑ 中指定的 BCD4 位数据与㉒ 中指定的 BCD4 位数据进行减法运算，将减法运算结果存储到 ⓓ 中指定的变量中。



(2) ㉑ , ㉒ , ⓓ 中可在 0 ～ 9999(BCD4 位) 的范围内进行指定。

(3) 减法结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



出 错

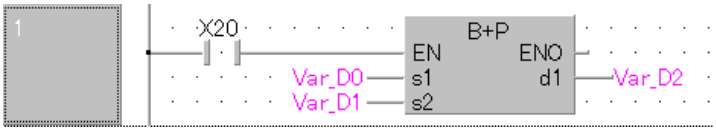
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

•㉑ , ㉒ , ⓓ 的 BCD 数据超出了 0 ～ 9999 的范围时。 (出错代码 : 4100)

程序示例

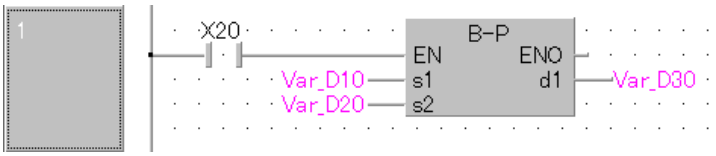
(1) 以下为 X20 变为 ON 时，将 Var_D0 的 BCD 数据与 Var_D1 的 BCD 数据进行加法运算，将结果存储到 Var_D2 中的程序。

[结构体梯形图]



(2) 以下为 X20 变为 ON 时，将 Var_D10 的 BCD 数据与 Var_D20 的 BCD 数据进行减法运算，将结果存储到 Var_D30 中的程序。

[结构体梯形图]



6.2.6 BCD8 位加减法

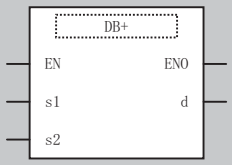
DB+, DB-

Basic | High performance | Universal | L CPU

DB+(P)
DB-(P)

P: 执行条件 :

结构体梯形图



ST

不支持。

中放入下述指令。

DB+	DB+P
DB-	DB-P

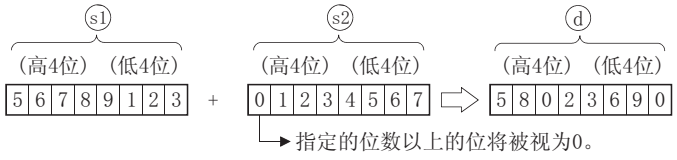
输入自变量，	EN:	执行条件	: 位
	s1:	被进行加减法的数据	: ANY32
	s2:	加减法数据	: ANY32
输出自变量，	ENO:	执行结果	: 位
	d:	运算结果	: ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ				○				-	-

功 能

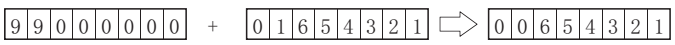
DB+(P)

(1) 将ⓢ1中指定的 BCD8 位数据与ⓢ2中指定的 BCD8 位数据进行加法运算，将加法结果存储到ⓓ中指定的变量中。



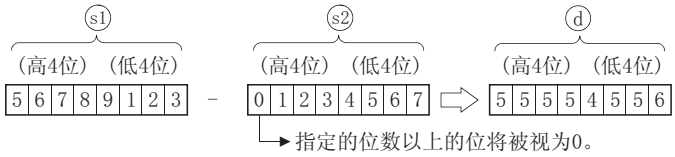
(2) ⓢ1, ⓢ2, ⓓ中可在 0 ～ 99999999(BCD8 位) 的范围内进行指定。

(3) 加法运算结果超过了 99999999 的情况下，进位将被忽略。
在这种情况下，进位标志不变为 ON。



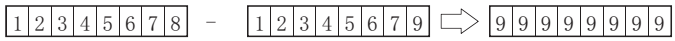
DB-(P)

(1) 将㉀ 中指定的 BCD 数据与㉁ 中指定的 BCD 数据进行减法运算，将减法运算结果存储到ⓓ 中指定的变量中。



(2) ㉀， ㉁， ⓓ 中可在 0 ～ 99999999(BCD8 位) 的范围内进行指定。

(3) 减法运算结果中发生了下溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。



出 错

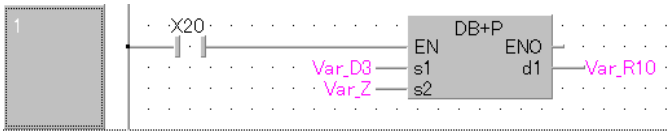
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

•㉀， ㉁， ⓓ 的 BCD 数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)

程序示例

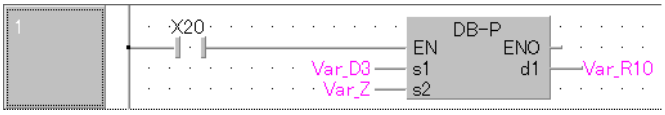
(1) 以下为 X20 为 ON 时，将 Var_D3 的 BCD 数据与 Var_Z 的 BCD 数据进行加法运算，将结果存储到 Var_R10 中的程序。

[结构体梯形图]



(2) 以下为 X20 为 ON 时，将 Var_D3 的 BCD 数据与 Var_Z 的 BCD 数据进行减法运算，将结果存储到 Var_R10 中的程序。

[结构体梯形图]



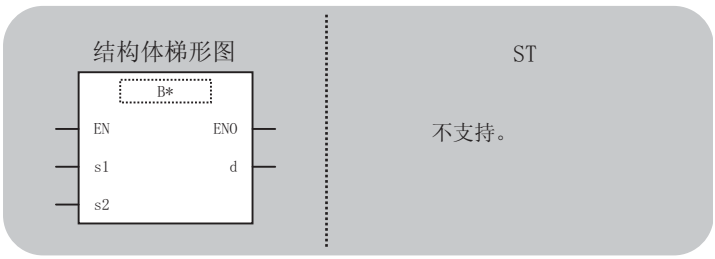
6.2.7 BCD4 位乘除法

B*, B/

Basic | High performance | Universal | L CPU

B*(P)
B/(P)

(P: 执行条件 : ↓)



中放入下述指令。

B*	B*P
B/	B/P

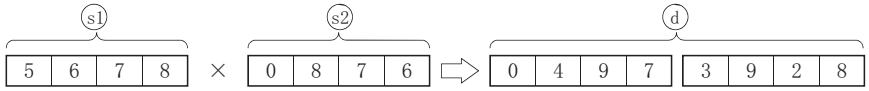
输入自变量， EN: 执行条件 : 位
s1: 被进行乘除法的数据或存储被进行乘除的数据的软元件的起始编号 : ANY16
s2: 乘除法数据或存储乘除法数据的软元件的起始编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 运算结果 : ANY32, ANY16 的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ				○				-	-

功 能

B*(P)

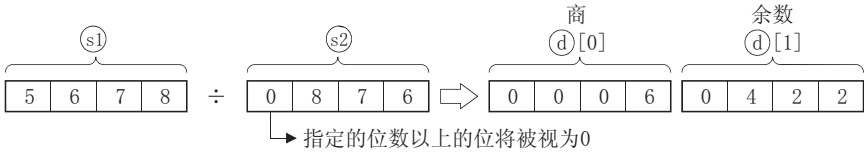
(1) 将ⓢ1 中指定的变量的 BCD 数据与ⓢ2 中指定的变量的 BCD 数据进行乘法运算，将结果存储到ⓓ 中指定的变量中。



(2) ⓢ1, ⓢ2 中可在 0 ～ 9999(BCD4 位) 的范围内进行指定。

B/(P)

(1) 将ⓢ1 中指定的变量的 BCD 数据与ⓢ2 中指定的变量的 BCD 数据进行除法运算，将结果存储到ⓓ 中指定的变量中。



(2) 对于除法运算结果，使用 32 位存储商及余数。

商 (BCD4 位)..... 被存储到ⓓ [0](16 位) 中。

余数 (BCD4 位) ..... 被存储到ⓓ [1](16 位) 中。

出 错

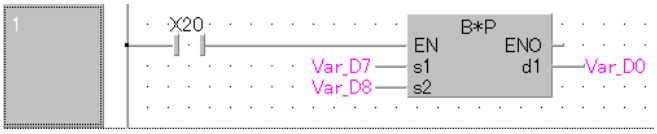
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1 ， ⓢ2 的 BCD 数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)
- 除数ⓢ2 为 0 时 (出错代码： 4100)

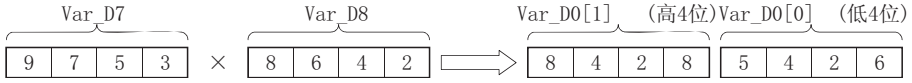
程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D7 的 BCD 数据与 Var_D8 的 BCD 数据进行乘法运算，将其结果存储到 Var_D0 中的程序。

[结构体梯形图]

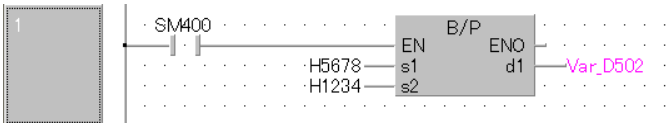


[动作]

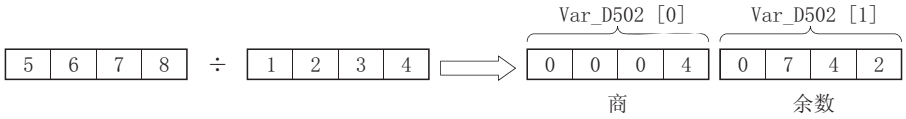


(2) 以下为将 5678 与 1234 的 BCD 数据进行除法运算，将其结果存储到 Var_D502 中的程序。

[结构体梯形图]



[动作]



6.2.8 BCD8 位乘除法

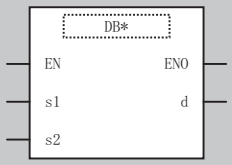
DB*, DB/

Basic | High performance | Universal | L CPU

DB*(P)
DB/(P)

P: 执行条件 :

结构体梯形图



ST

不支持。

中放入下述指令。

DB*	DB*P
DB/	DB/P

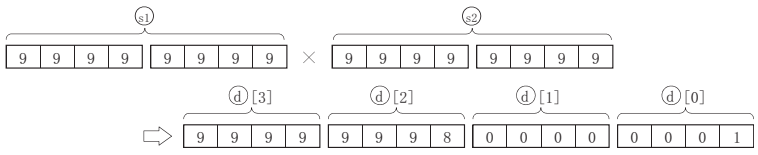
输入自变量，	EN:	执行条件	: 位
	s1:	被进行乘除法的数据	: ANY32
	s2:	乘除法数据	: ANY32
输出自变量，	ENO:	执行结果	: 位
	d:	运算结果	: ANY16 的数组 (0..3), ANY32 的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	○			○					-
ⓢ2	○			○					-
ⓓ	○			-					-

功 能

DB*(P)

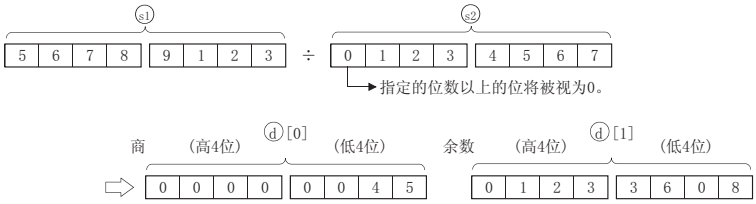
(1) 将ⓢ1 中指定的变量的 BCD8 位数据与ⓢ2 中指定的变量的 BCD8 位数据进行乘法运算，将乘法结果存储到ⓓ 中指定的变量中。



(2) ⓢ1, ⓢ2 中可在 0 ～ 99999999(BCD8 位) 的范围内进行指定。

DB/(P)

(1) 将ⓢ1 中指定的 BCD8 位数据与ⓢ2 中指定的 BCD8 位数据进行除法运算，将结果存储到ⓓ 中指定的变量中。



(2) 对于除法运算结果，使用 64 位存储商及余数。

商 (BCD8 位)..... 被存储到ⓓ [0](32 位) 中。

余数 (BCD8 位) .... 被存储到ⓓ [1](32 位) 中。

出 错

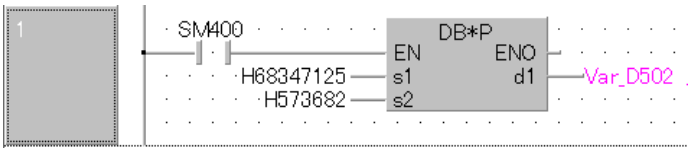
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1 , ⓢ2 的 BCD 数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)
- 除数ⓢ2 为 0 时。 (出错代码： 4100)

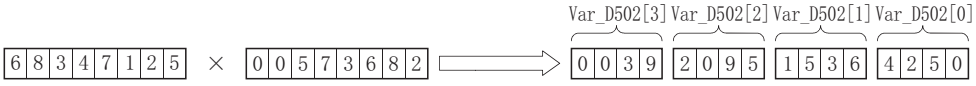
程序示例

(1) 以下为将 68347125 与 573682 的 BCD 数据进行乘法运算，将其结果存储到 Var_D502 中的程序。

[结构体梯形图]

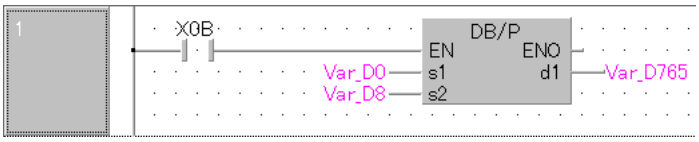


[动作]

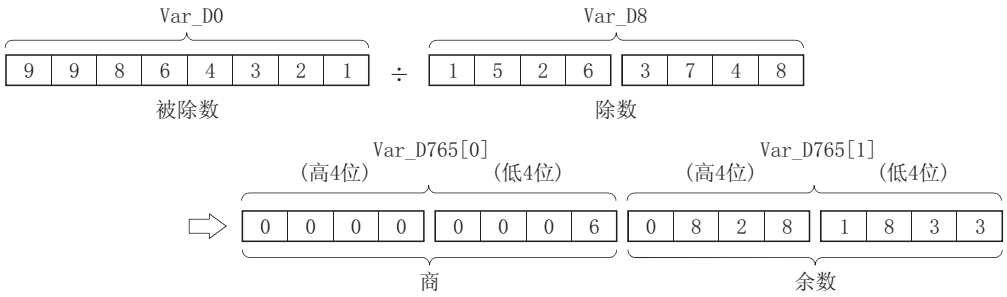


(2) 以下为 X0B 变为 ON 时，将 Var_D0 的 BCD 数据与 Var_D8 的 BCD 数据进行除法运算，将其结果存储到 Var_D765 中的程序。

[结构体梯形图]



[动作]



6.2.9 浮动小数点数据加减法（单精度）

E+, E-

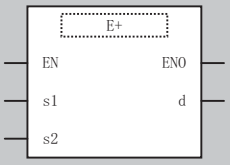
Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

E+(P)
E-(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

中放入下述指令。

E+	E+P
E-	E-P

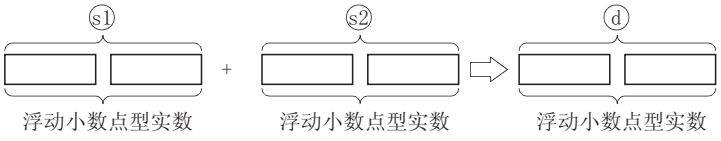
输入自变量，EN：　执行条件　：位
　　s1：　被进行加减法的数据　：单精度实数
　　s2：　加减法数据　：单精度实数
输出自变量，ENO：　执行结果　：位
　　d：　运算结果　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ1	-	○		-	○		-	○	-
ⓢ2	-	○		-	○		-	○	-
ⓓ	-	○		-	○		-	-	-

功 能

E+(P)

(1) 将ⓢ1 中指定的浮动小数点型实数与ⓢ2 中指定的浮动小数点型实数进行加法运算，将加法运算结果存储到ⓓ 中指定的变量中。

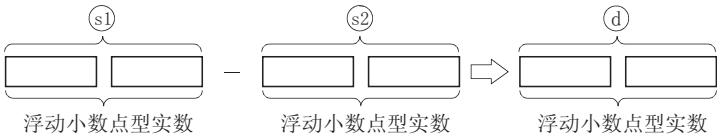


(2) ⓢ1，ⓢ2，ⓓ 中可指定的值及可存储的值如下所示。

0，$2^{-126} \leqq |$ 指定值（存储值）$| < 2^{128}$

E-(P)

(1) 将$\textcircled{s1}$中指定的浮动小数点型实数与$\textcircled{s2}$中指定的浮动小数点型实数进行减法运算，将减法运算结果存储到$\textcircled{d}$中指定的变量中。



(2) $\textcircled{s1}$，$\textcircled{s2}$，$\textcircled{d}$中可指定的值及可存储的值如下所示。

$0,\ 2^{-126} \leqq |$指定值（存储值）$| < 2^{128}$

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。
  $0,\ 2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$
  （基本型 QCPU、高性能型 QCPU 时）　（出错代码：4100）
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时）　（出错代码：4100）

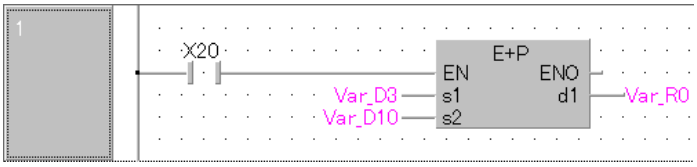
*1：　有的 CPU 模块即使被指定为 -0 也不会变为运算出错状态。
详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 运算结果超出了下述范围时。（发生了上溢时。）（通用型 QCPU、LCPU 时）
  $2^{128} \leqq |$运算结果$|$　（出错代码：4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）　（出错代码：4140）

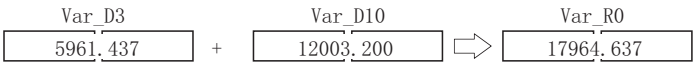
程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D3 的浮动小数点型实数与 Var_D10 的浮动小数点型实数进行加法运算，将加法结果存储到 Var_R0 中的程序。

[结构体梯形图]

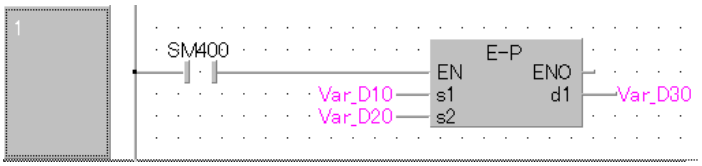


[动作]

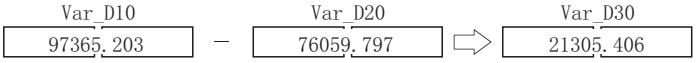


(2) 以下为将 Var_D10 的浮动小数点型实数与 Var_D20 的浮动小数点型实数进行减法运算，将减法结果存储到 Var_D30 中的程序。

[结构体梯形图]



[动作]



6.2.10 浮动小数点数据加减法（双精度）

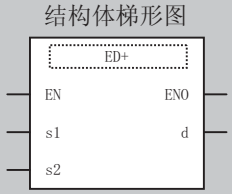
ED+, ED-

Basic | High performance | Universal | L CPU

ED+(P)
ED-(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

ED+	ED+P
ED-	ED-P

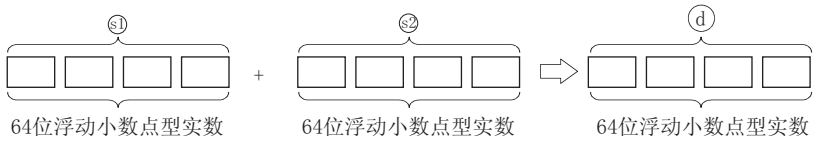
输入自变量，	EN:	执行条件	：位
	s1:	被进行加减法的数据	：双精度实数
	s2:	加减法数据	：双精度实数
输出自变量，	ENO:	执行结果	：位
	d:	运算结果	：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ1	-	○		-				○	-
ⓢ2	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

ED+(P)

(1) 将ⓢ1 中指定的 64 位浮动小数点型实数数据与ⓢ2 中指定的 64 位浮动小数点型实数数据进行加法运算，将加法结果存储到ⓓ 中指定的变量中。

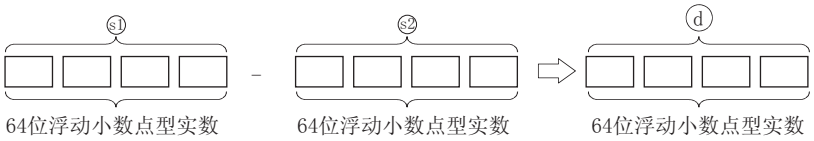


(2) ⓢ1，ⓢ2，ⓓ 中可指定的值及可存储的值如下所示。

0，　$2^{-1022} \leqq |$ 指定值（存储值）$| < 2^{1024}$

ED-(P)

(1) 将$S1$中指定的 64 位浮动小数点型实数数据与$S2$中指定的 64 位浮动小数点型实数数据进行减法运算，将减法结果存储到d中。



(2) $S1$，$S2$，d中可指定的值及可存储的值如下所示。

0，　$2^{-1022} \leqq |$ 指定值（存储值）$| < 2^{1024}$

出 错

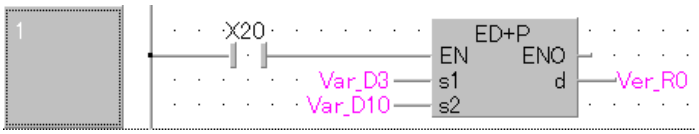
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。　（出错代码： 4140）
  0，$2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。　（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$　（出错代码： 4141）

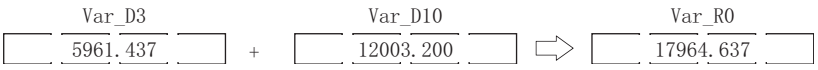
程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D3 的 64 位浮动小数点型实数与 Var_D10 的 64 位浮动小数点型实数进行加法运算，将加法结果存储到 Var_R0 中的程序。

[结构体梯形图]

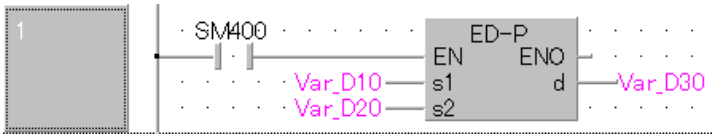


[动作]

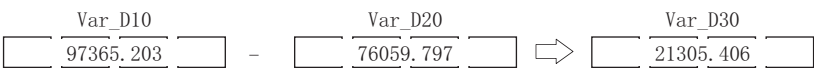


(2) 以下为将 Var_D10 的 64 位浮动小数点型实数与 Var_D20 的 64 位浮动小数点型实数进行减法运算，将减法结果存储到 Var_D30 中的程序。

[结构体梯形图]



[动作]



6.2.11 浮动小数点数据乘除法（单精度）

E*, E/

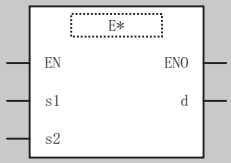
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

E*(P)
E/(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

E*	E*P
E/	E/P

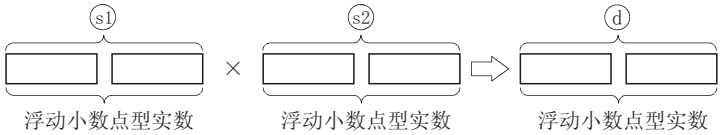
输入自变量，EN：　执行条件　：位
　　s1：　被进行乘除法的数据　：单精度实数
　　s2：　乘除法数据　：单精度实数
输出自变量，ENO：　执行结果　：位
　　d：　运算结果　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ1	-	○		-	○		-	○	-
ⓢ2	-	○		-	○		-	○	-
ⓓ	-	○		-	○		-	-	-

功 能

E*(P)

(1) 将ⓢ1 中指定的浮动小数点型实数与ⓢ2 中指定的浮动小数点型实数进行乘法运算，将乘法结果存储到ⓓ 中指定的变量中。

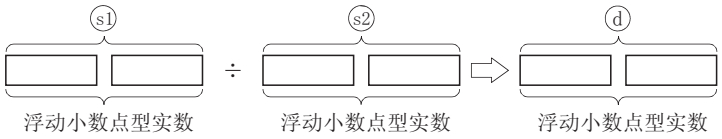


(2) ⓢ1，ⓢ2，ⓓ 中可指定的值及可存储的值如下所示。

0，$2^{-126} \leqq$ | 指定值（存储值）| $< 2^{128}$

E/(P)

(1) 将ⓢ1 中指定的浮动小数点型实数与ⓢ2 中指定的浮动小数点型实数进行除法运算，将除法运算结果存储到ⓓ 中指定的变量中。



(2) ⓢ1，ⓢ2，ⓓ 中可指定的值及可存储的值如下所示。

0，$2^{-126} \leqq |$ 指定值（存储值）$| < 2^{128}$

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。
  0，$2^{-126} \leqq |$ 指定软元件的内容 $| < 2^{128}$
  （基本型 QCPU、高性能型 QCPU 时）　（出错代码：4100）
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时）　（出错代码：4100）

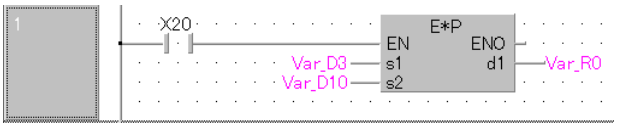
*1：　有的 CPU 模块即使被指定为 -0 也不会变为运算出错状态。
　　详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 运算结果超出了下述范围时。（发生了上溢时。）（通用型 QCPU、LCPU 时）
  $2^{128} \leqq |$ 运算结果 $|$　（出错代码：4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）　（出错代码：4140）

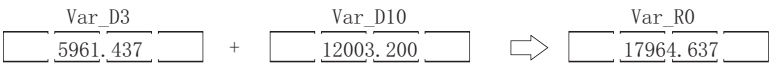
程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D3 的浮动小数点型实数与 Var_D10 的浮动小数点型实数进行乘法运算，将乘法运算结果存储到 Var_R0 中的程序。

[结构体梯形图]

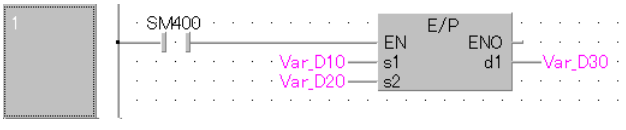


[动作]

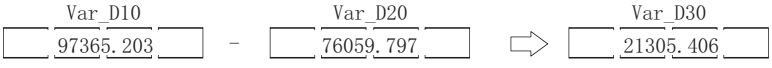


(2) 以下为将 Var_D10 的浮动小数点型实数与 Var_D20 的浮动小数点型实数进行除法运算，将除法运算结果存储到 Var_D30 中的程序。

[结构体梯形图]



[动作]



6.2.12 浮动小数点数据乘除法（双精度）

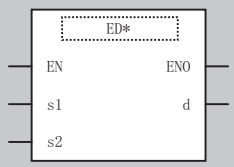
ED*, ED/

Basic | High performance | Universal | L CPU

ED*(P)
ED/(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

ED*　　ED*P
ED/　　ED/P

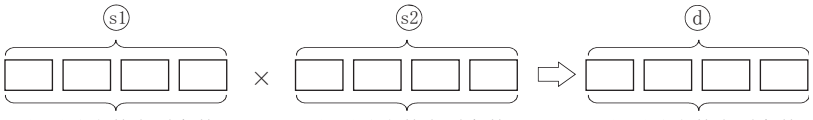
输入自变量，EN:　执行条件　：位
　　s1:　被进行乘除法的数据　：双精度实数
　　s2:　乘除法数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ1	-	○		-				○	-
ⓢ2	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

ED*(P)

(1) 将ⓢ1 中指定的 64 位浮动小数点型实数数据与ⓢ2 中指定的 64 位浮动小数点型实数数据进行乘法运算，将乘法结果存储到ⓓ 中指定的变量中。



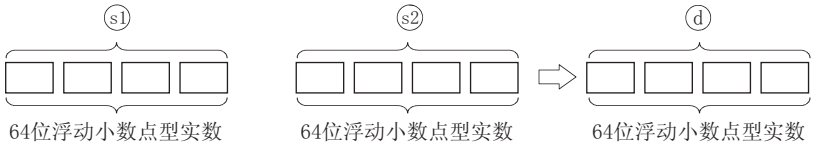
(2) ⓢ1，ⓢ2，ⓓ 中可指定的值及可存储的值如下所示。

0，$2^{-1022} \leqq |$指定值（存储值）$| < 2^{1024}$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

ED/(P)

(1) 将㉑中指定的 64 位浮动小数点型实数数据与㉒中指定的 64 位浮动小数点型实数数据进行除法运算，将除法运算结果存储到ⓓ中。



(2) ㉑，㉒，ⓓ中可指定的值及可存储的值如下所示。

$0,\ 2^{-1022} \leqq |$ 指定值（存储值）$| < 2^{1024}$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

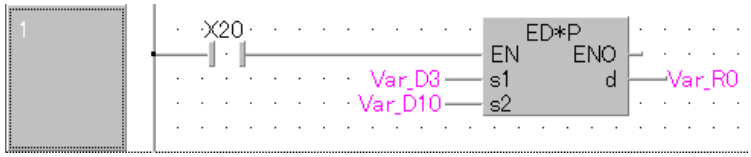
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。
  $0,\ 2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$ （出错代码： 4140）
- 指定软元件的内容为 -0 时。 （出错代码： 4140）
- 除法运算指令时㉒内容为 0 时。 （出错代码： 4100）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$ （出错代码： 4141）

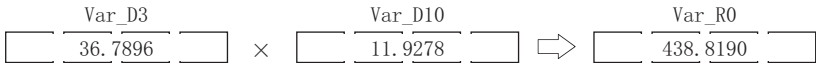
程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D3 的 64 位浮动小数点型实数与 Var_D10 的 64 位浮动小数点型实数进行乘法运算，将乘法运算结果存储到 Var_R0 中的程序。

[结构体梯形图]

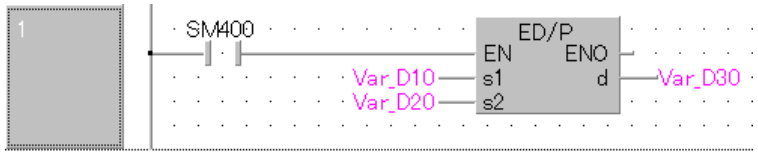


[动作]

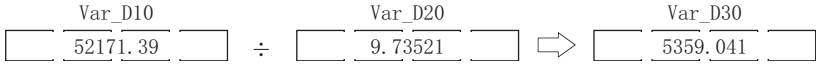


(2) 以下为将 Var_D10 的 64 位浮动小数点型实数与 Var_D20 的 64 位浮动小数点型实数进行除法运算，将除法运算结果存储到 Var_D30 中的程序。

[结构体梯形图]



[动作]



6.2.13 BIN16 位数据块加减法

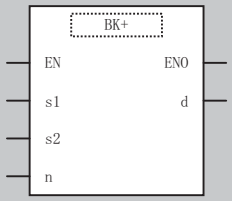
BK+, BK-

Basic | High performance | Universal | L CPU

BK+(P)
BK-(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

中放入下述指令。

BK+　　BK+P
BK-　　BK-P

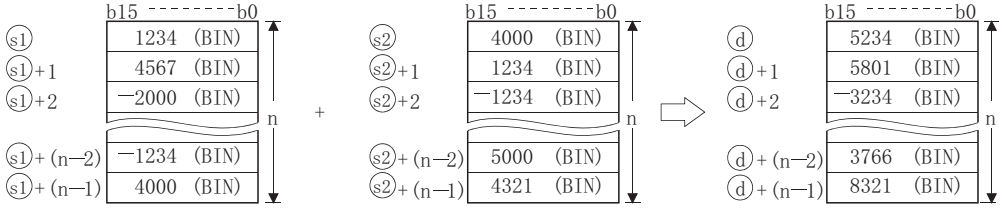
输入自变量，EN：　执行条件　：位
　　s1：　存储被进行加减法的数据的软元件的起始编号　：ANY16
　　s2：　存储加减法数据的软元件的起始编号　：ANY16
　　n：　加减法数据个数　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	-	○		-				-	-
ⓢ2	-	○		-				○	-
n	○	○		○				○	-
ⓓ	-	○		-				-	-

功 能

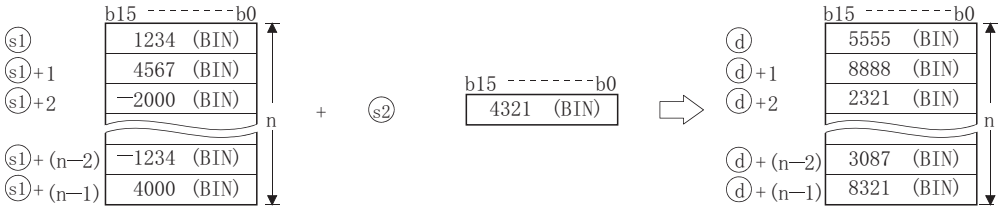
BK+(P)

(1) 将ⓢ1 中指定的软元件算起 n 点的 BIN 数据与ⓢ2 中指定的软元件算起 n 点的 BIN 数据进行加法运算，将运算结果存储到ⓓ 中指定的软元件以后。



(2) 块加法运算是以 16 位为单位进行。

(3) ⓢ2 中可以指定 -32768 ～ 32767(BIN16 位) 的常数。



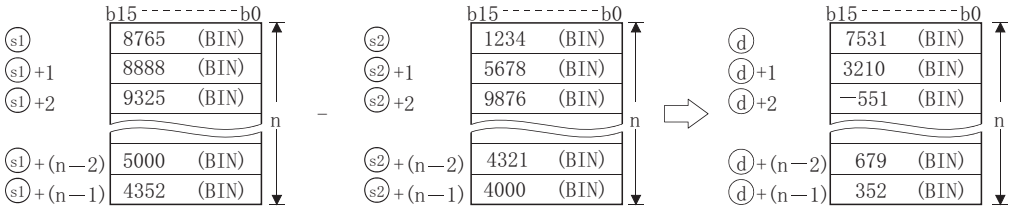
(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- 32767 +2 ⟶ −32767
  (7FFFH) (0002H) (8001H)

- −32767 +−2 ⟶ 32767
  (8001H) (FFFEH) (7FFFH)

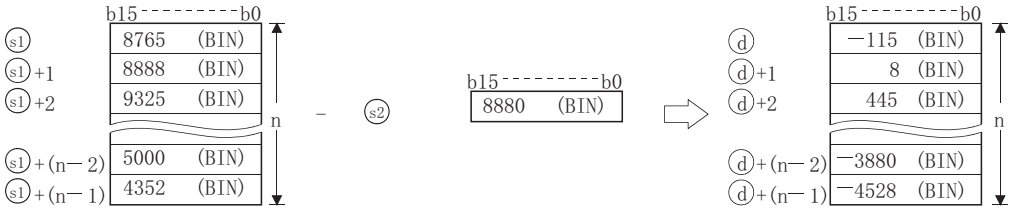
BK-(P)

(1) 将ⓢ1 中指定的软元件算起 n 点的 BIN 数据与ⓢ2 中指定的软元件算起 n 点的 BIN 数据进行减法运算，将运算结果存储到ⓓ 中指定的软元件以后。



(2) 块减法运算是以 16 位为单位进行。

(3) ⓢ2 中可以指定 -32768 ～ 32767(BIN16 位) 的常数。



(4) 运算结果中发生了下溢 / 上溢时的情况如下所示。
在这种情况下，进位标志不变为 ON。

- −32768 −2 ⟶ 32766
  (8000H) (0002H) (7FFEH)

- 32767 − −2 ⟶ −32767
  (7FFFH) (FFFEH) (8001H)

出 错

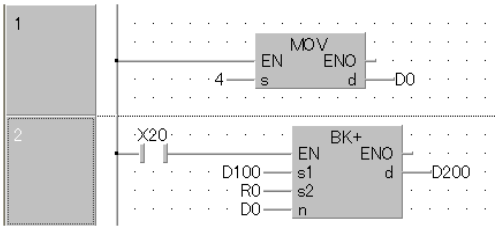
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1，ⓢ2，ⓓ 的软元件算起 n 点的范围超出了相应软元件时。
  （出错代码： 4101）
- ⓢ1 与ⓓ 的软元件范围重叠时。(ⓢ1 与ⓓ 中指定为同一软元件时除外。)
  （出错代码： 4101）
- ⓢ2 与ⓓ 的软元件范围重叠时。(ⓢ2 与ⓓ 中指定为同一软元件时除外。)
  （出错代码： 4101）

程序示例

(1) 以下为 X20 变为 ON 时，将 D100 ～ D103 中存储的值与 R0 ～ R3 中存储的值进行加法运算，将其结果存储到 D200 以后的程序。

[结构体梯形图]



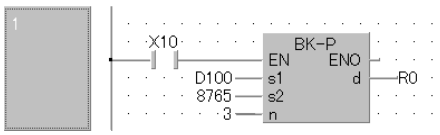
[动作]

	b15---------b0			b15---------b0			b15---------b0
D100	6789 (BIN)		R0	1234 (BIN)		D200	8023 (BIN)
D101	7821 (BIN)	+	R1	2032 (BIN)	⇨	D201	9853 (BIN)
D102	5432 (BIN)		R2	−3252 (BIN)		D202	2180 (BIN)
D103	3520 (BIN)		R3	−1000 (BIN)		D203	2520 (BIN)

D0 4

(2) 以下为 X1C 变为 ON 时，将 D100 ～ D102 的数据与常数 8765 进行减法运算，将其结果存储到 R0 以后的程序。

[结构体梯形图]



[动作]

	b15---------b0		b15---------b0			b15---------b0
D100	12345 (BIN)				R0	3580 (BIN)
D101	8701 (BIN)	−	8765 (BIN)	⇨	R1	−64 (BIN)
D102	3502 (BIN)				R2	−5263 (BIN)

6.2.14 BIN32 位数据块加减法

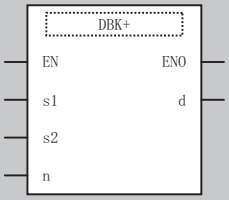
DBK+, DBK-

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

DBK+(P)
DBK-(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

中放入下述指令。

DBK+	DBK+P
DBK-	DBK-P

输入自变量，EN：　执行条件　：位
　　s1：　存储被进行加减法的数据的软元件的起始编号　：ANY32
　　s2：　加减法数据或存储加减法数据的软元件的起始编号　：ANY32
　　n：　加减法数据个数　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储运算结果的软元件的起始编号　：ANY32

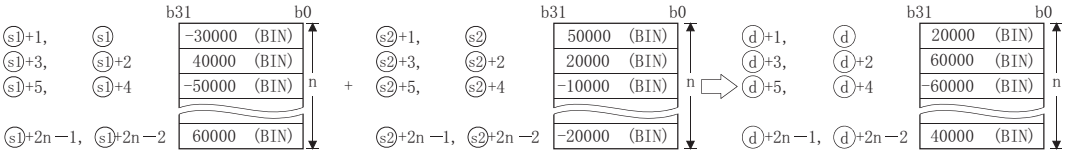
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	－	○		－				－	－
ⓢ2	－	○		－				○	－
n	○	○		○				○	－
ⓓ	－	○		－				－	－

功 能

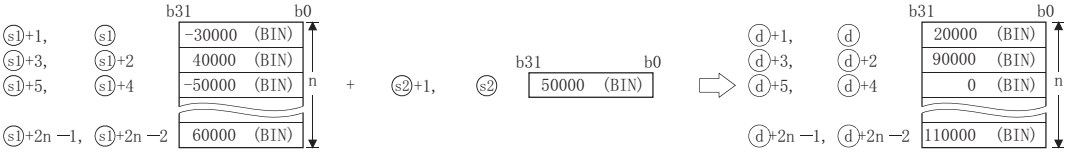
DBK+

(1) 将㉑ 中指定的软元件算起 n 点的 BIN32 位数据与㉒ 中指定的软元件算起 n 点的 BIN32 位数据或常数进行加法运算，将运算结果存储到ⓓ 中指定的软元件以后。

•㉒ 中指定了软元件的情况



•㉒ 中指定了常数的情况



(2) 块加法运算是以 32 位为单位进行。

(3) ㉒ 中可以指定 -2147483648 ～ 2147483647(BIN32 位) 的常数。

(4) n 中指定的值为 0 时将变为无处理。

(5) 运算结果中发生了上溢时的情况如下所示。
在这种情况下，进位标志也不变为 ON。

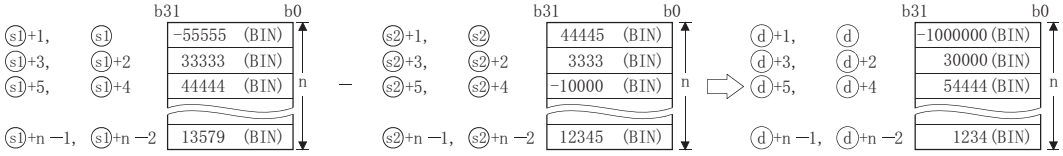
- K2147483647 +K2 ⟶ K－2147483647
  (7FFFFFFF$_H$) (00000002$_H$) (80000001$_H$)

- K－2147483647+K－2 ⟶ K2147483647
  (80000001$_H$) (FFFFFFFE$_H$) (7FFFFFFF$_H$)

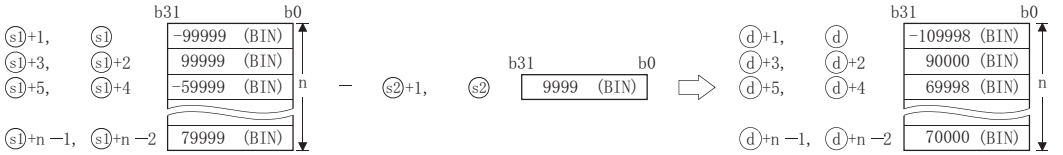
DBK-

(1) 将㉑ 中指定的软元件算起 n 点的 BIN32 位数据与㉒ 中指定的软元件算起 n 点的 BIN32 位数据或常数进行减法运算，将运算结果存储到ⓓ 中指定的软元件以后。

•㉒ 中指定了软元件的情况



•㉒ 中指定了常数的情况



(2) 块减法运算是以 32 位为单位进行。

(3) ㉒ 中可以指定 -2147483648 ～ 2147483647(BIN32 位) 的常数。

(4) n 中指定的值为 0 时将变为无处理。

(5) ⓓ 的指定应在㉑ 算起 n 点的软元件范围及㉒ 算起 n 点的软元件范围以外。
但是，可以指定与㉑ 或㉒ 相同的软元件。

(6) 运算结果中发生了上溢时的情况如下所示。
在这种情况下，进位标志也不变为 ON。

- K2147483647 − K−2 ⟶ K−2147483647
  ($7FFFFFFF_H$) (00000002_H) (80000001_H)

- K−2147483647 −K2 ⟶ K2147483647
  (80000001_H) ($FFFFFFFE_H$) ($7FFFFFFF_H$)

出 错

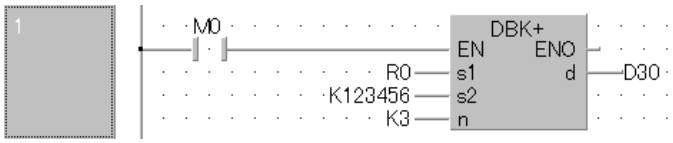
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 中指定了负值时。 (出错代码： 4100)
- ㉑ , ㉒ , ⓓ 中指定的软元件算起的 n 点超出了指定软元件的范围时。
  (出错代码： 4101)
- ㉑ 算起 n 点的软元件范围与ⓓ 算起 n 点的软元件范围重叠时。
  (㉑ 与ⓓ 中指定了同一软元件时除外。) (出错代码： 4101)
- ㉒ 算起 n 点的软元件范围与ⓓ 算起 n 点的软元件范围重叠时。 (出错代码： 4101)

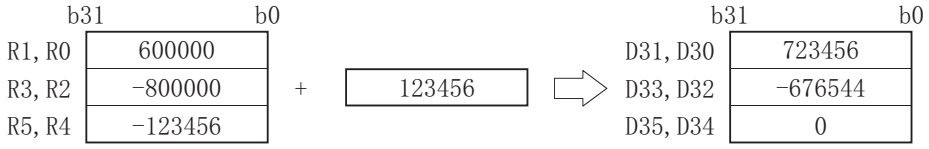
程序示例

(1) 以下为 M0 变为 ON 时，将 R0 ～ R5 中存储的值与常数进行加法运算，将运算结果存储到 D30 ～ D35 中的程序。

[结构体梯形图]

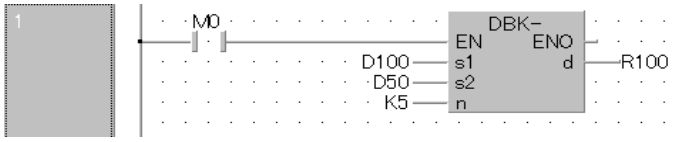


[动作]

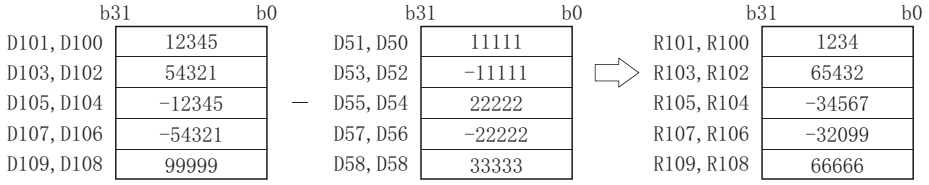


(2) 以下为 M0 变为 ON 时，将 D100 ～ D109 中存储的值与 D50 ～ D59 中存储的值进行减法运算，将运算结果存储到 R100 ～ R109 中的程序。

[结构体梯形图]



[动作]



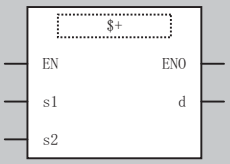
6.2.15 字符串的合

$+

Basic | High performance | Universal | L CPU

$+(P)	P：执行条件　　：↑

结构体梯形图



ST

不支持。

□ 中放入下述指令。

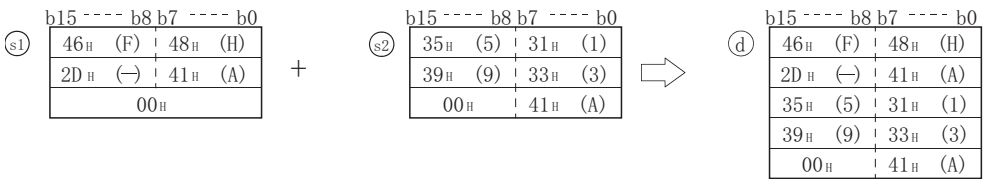
$+　　$+P

输入自变量，EN：执行条件　：位
　　s1：连接数据　：字符串
　　s2：被连接的数据　：字符串
输出自变量，ENO：执行结果　：位
　　d：栝寂寂壏　：字符串

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
⑤1	－	○		－				○	－
⑤2	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 在⑤1 中指定的字符串数据的后面，连接⑤2 中指定的字符串数据，将结果存储到ⓓ 中指定的变量中。



(2) 进行字符串的合并时，表示⑤1 中指定的字符串的结束的 00H 将被忽略，在⑤1 的最终字符处连接⑤2 指定的字符串。

出 错

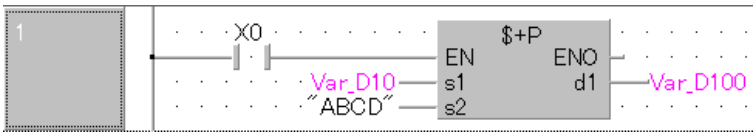
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 中指定的变量中不能存储全部的字符串时。 （出错代码： 4101）
- ⓢ1 与ⓢ2 中指定的变量重复时。 （出错代码： 4101）
- ⓢ2 与ⓓ 中指定的变量重复时。 （出错代码： 4101）
- ⓢ1， ⓢ2， ⓓ 的字符串超过了 16383 个字符时。 （出错代码： 4101）

程序示例

以下为 X0 变为 ON 时，将 Var_D10 中存储的字符串与字符串 “ABCD”合并，存储到 Var_D100 以后的程序。

[结构体梯形图]



6.2.16 16 位 BIN 数据递增、递减

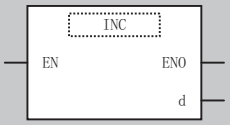
INC, DEC

Basic | High performance | Universal | L CPU

INC(P)
DEC(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= INC (EN, d);

[ ] 中放入下述指令。

INC	INCP
DEC	DECP

输入自变量, EN: 执行条件 : 位
输出自变量, ENO: 执行结果 : 位
d: INC(+1),DEC(-1) 的变量 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ			○					-	

功 能

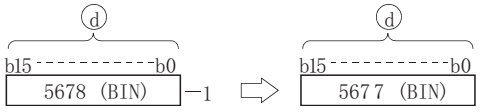
INC(P)

(1) 将ⓓ 中指定的变量进行 (16 位数据)+1。

(2) ⓓ 中指定的变量的内容为 32768 时执行的情况下，在指定的变量中将存储 -32768。

DEC(P)

(1) ⓓ 中指定的变量进行 (16 位数据)-1。



(2) ⓓ 中指定的变量的内容为 -32768 时执行的情况下，在指定的变量中将存储 32767。

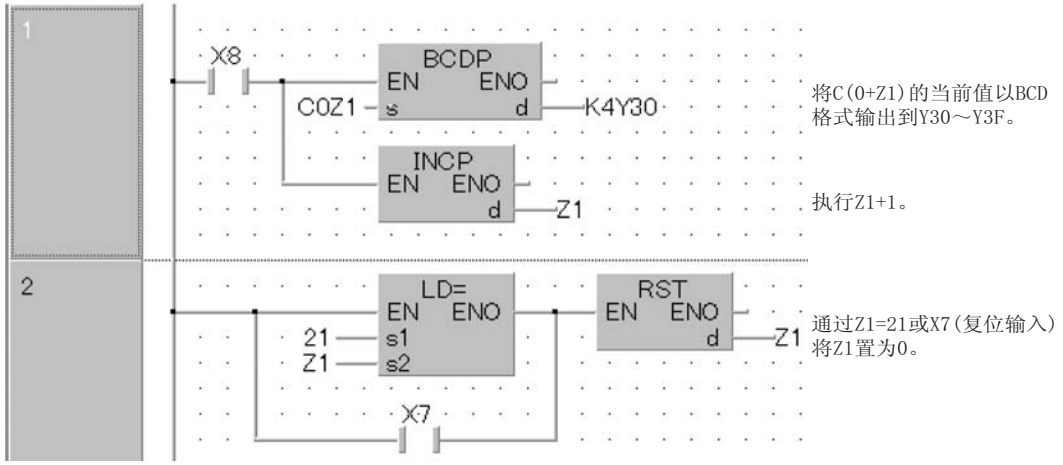
出 错

不存在 INC(P)/DEC(P) 指令相关的运算出错。

程序示例

(1) 以下为 X8 为 ON 时，将计数器 C0 ～ C20 的当前值以 BCD 格式输出到 Y30 ～ Y3F 中的程序。
(当前值＜ 9999 时)

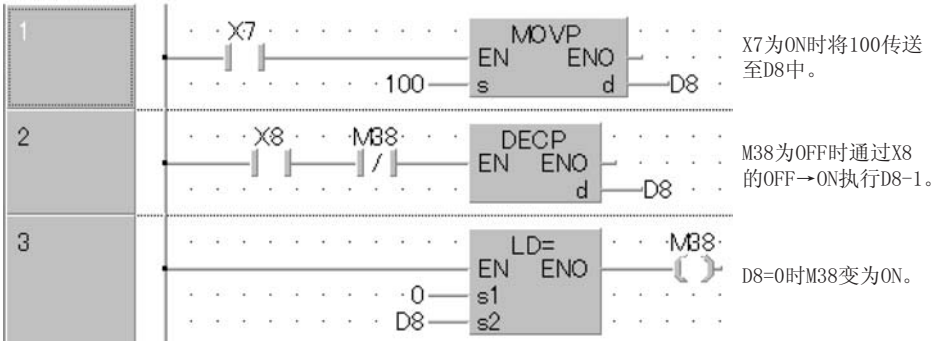
[结构体梯形图]




```
[ST]
BCDP(X8,C0Z1,K4Y30);
INCP(X8,Z1);
IF Z1=21 OR X7 THEN
     RST(TRUE,Z1);
END_IF;
```

(2) 减法计数器的程序。

[结构体梯形图]




```
[ST]
MOVP(X7,100,D8);
IF X8 AND NOT(M38) THEN
     DECP(TRUE,D8);
END_IF;
OUT(D8=0,M38);
```

6.2.17 32 位 BIN 数据递增、递减

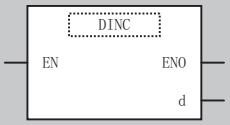
DINC, DDEC

Basic | High performance | Universal | L CPU

DINC(P)
DDEC(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= DINC (EN, d);
```

[ ] 中放入下述指令。

DINC	DINCP
DDEC	DDECP

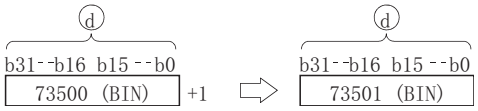
输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: 进行 DINC(+1).DDEC(-1) 的变量 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ			○					-	

功 能

DINC(P)

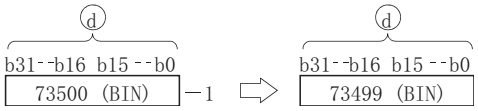
(1) 将ⓓ 中指定的变量进行 (32 位数据)+1。



(2) ⓓ 中指定的变量的内容为 2147483647 时执行的情况下，在ⓓ 中指定的变量中将存储 -2147483648。

DDEC(P)

(1) 将ⓓ 中指定的变量进行 (32 位数据)-1。



(2) ⓓ 中指定的变量的内容为 0 时执行的情况下，在ⓓ 指定的变量中将存储 -1。

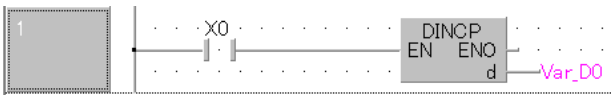
出 错

不存在 DINC(P)/DDEC(P) 指令相关的运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，对 Var_D0 的数据进行 +1 的程序。

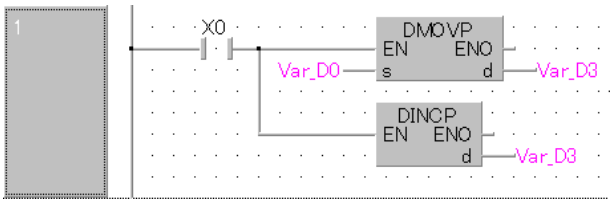
[结构体梯形图]




```
[ST]
DINCP(X0,Var_D0);
```

(2) 以下为 X0 变为 ON 时，对设置到 Var_D0 的数据进行 +1，将结果存储到 Var_D3 中的程序。

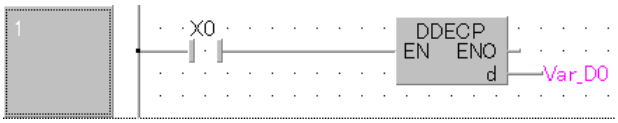
[结构体梯形图]




```
[ST]
IF X0 THEN
    DMOVP(TRUE,Var_D0,Var_D3);
    DINCP(TRUE,Var_D3);
END_IF;
```

(3) 以下为 X0 变为 ON 时，对 Var_D0 的数据进行 -1 的程序。

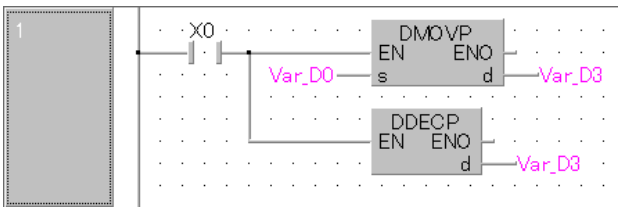
[结构体梯形图]




```
[ST]
DDECP(X0,Var_D0);
```

(4) 以下为 X0 变为 ON 时，对设置到 Var_D0 的数据进行 -1，将结果存储到 Var_D3 中的程序。

[结构体梯形图]




```
[ST]
IF X0 THEN
    DMOVP(TRUE,Var_D0,Var_D3);
    DDECP(TRUE,Var_D3);
END_IF;
```

6.3 数据转换指令

6.3.1 BIN 数据→ BCD4 位 /8 位转换

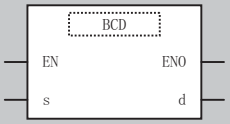
BCD, DBCD

Basic | High performance | Universal | L CPU

BCD(P)
DBCD(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= BCD (EN, s, d);
```

[ ] 中放入下述指令。

BCD	BCDP
DBCD	DBCDP

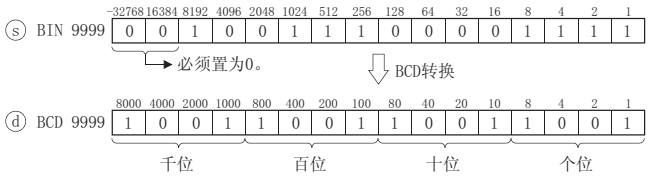
输入自变量， EN: 执行条件 : 位
s: BIN : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 转换后的 BCD 数据 : ANY16/32

设置数据	内部软元件 位	内部软元件 字	R, ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K, H	其它
ⓢ		○						○	-
ⓓ		○						-	-

功 能

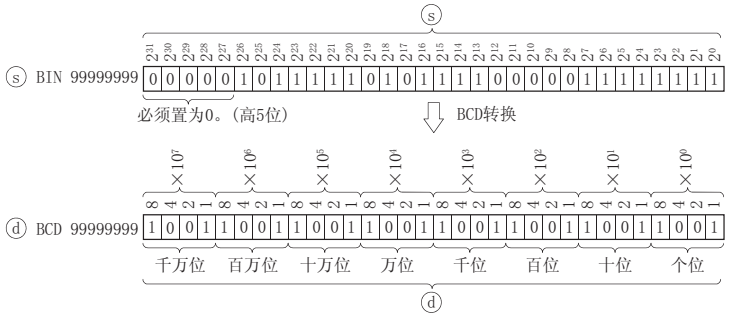
BCD(P)

将ⓢ 中指定的软元件的 BIN 数据 (0 ～ 9999) 进行 BCD 转换后，存储到ⓓ 中指定的软元件中。



DBCD(P)

将ⓢ中指定的软元件的 BIN 数据 (0 ～ 99999999) 进行 BCD 转换后，存储到ⓓ中指定的软元件中。



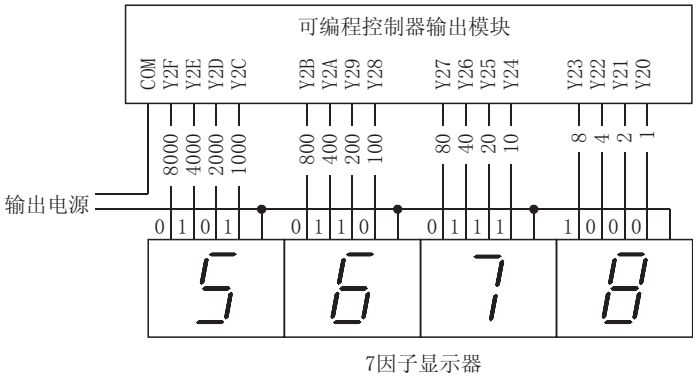
出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- BCD(P) 指令时ⓢ的数据超出了 0 ～ 9999 的范围时。 (出错代码： 4100)
- DBCD(P) 指令时ⓢ的数据超出了 0 ～ 99999999 的范围时。 (出错代码： 4100)

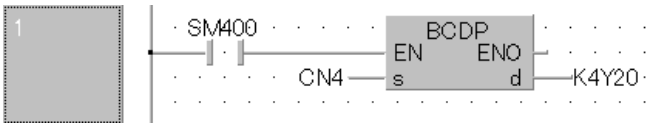
程序示例

(1) 以下为将 C4 的当前值从 Y20 ～ Y2F 输出到 BCD 显示器的程序。



7因子显示器

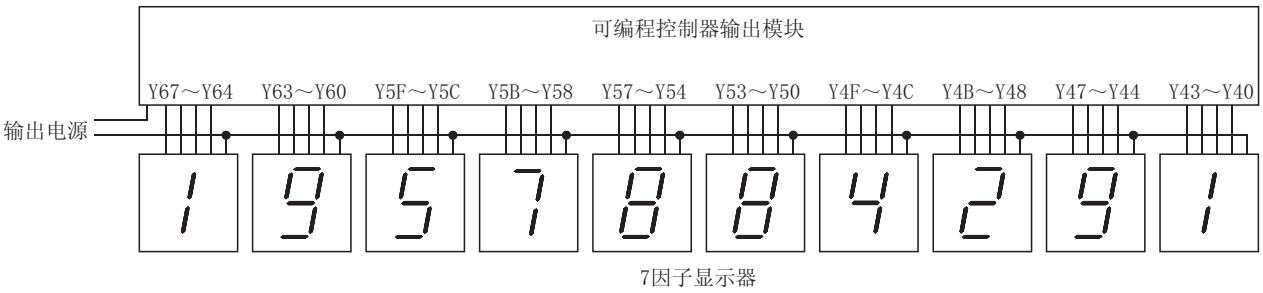
[结构体梯形图]



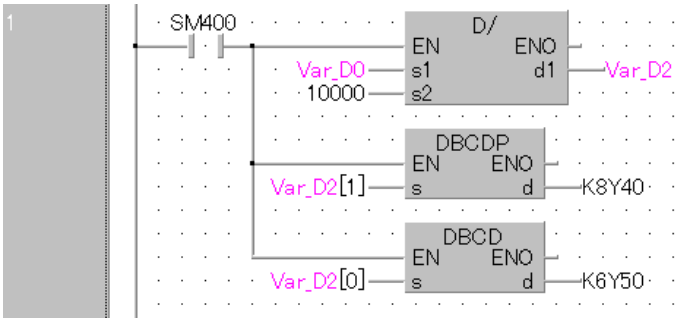
[ST]

```
BCDP(SM400,CN4,K4Y20);
```

(2) 以下为将 D0 ～ D1 的 32 位数据输出到 Y40 ～ Y67 中的程序。



[结构体梯形图]




```
[ST]
IF SM400 THEN
     Var_D2[0]:=Var_D0/10000;
     DBCDP(TRUE,Var_D2[1],K8Y40);
     DBCD(TRUE,Var_D2[0],K6Y50);
END_IF;
```

6.3.2 BCD4 位 /8 位 -BIN 转换

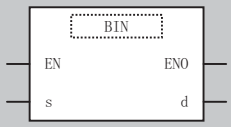
BIN, DBIN

Basic | High performance | Universal | L CPU

BIN(P)
DBIN(P)

P: 执行条件　　: ↵

结构体梯形图



ST

ENO:= BIN (EN, s, d);

[ ] 中放入下述指令。

BIN	BINP
DBIN	DBINP

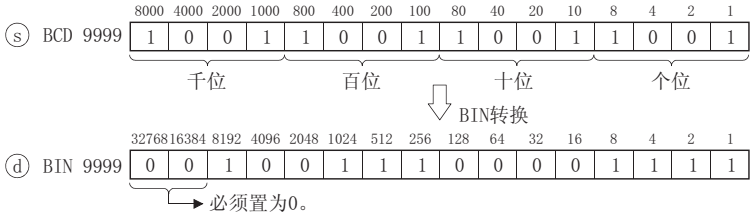
输入自变量，EN:　执行条件　：位
　　s:　BCD　：ANY16/32
输出自变量，ENO:　执行结果　：位
　　d:　转换后的 BIN 数据　：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○				○	-
ⓓ				○				-	-

功 能

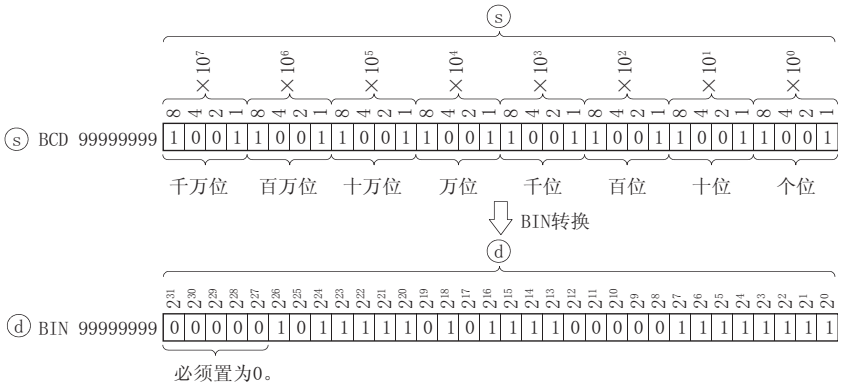
BIN(P)

将ⓢ 中指定的软元件的 BCD 数据 (0 ～ 9999) 进行 BIN 转换后，存储到ⓓ 中指定的软元件中。



DBIN(P)

将ⓢ中指定的软元件的 BCD 数据（0 ～ 99999999）进行 BIN 转换后，存储到ⓓ中指定的软元件中。



出 错

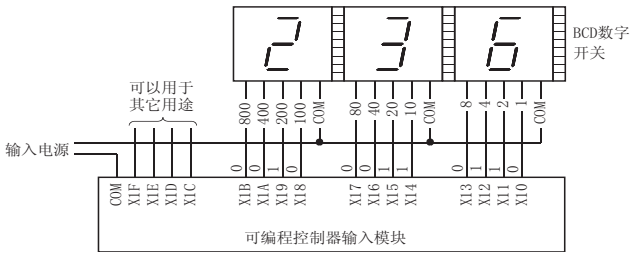
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ 的各位中有除 0 ～ 9 以外的值时。（出错代码： 4100）

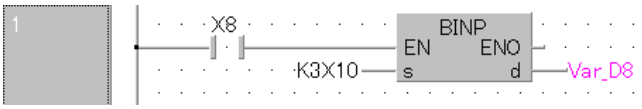
通过将 SM722 预先置为 ON，可以避免发生上述出错。
再者，设置了超出范围的数值的情况下，与 SM722 的 ON/OFF 状态无关，指令将被执行。
此外，BINP/DBINP 指令的情况下与出错的有无无关，在将指令（执行条件）置为 OFF → ON 之前不能执行下一个运算。

程序示例

(1) 以下为 X8 变为 ON 时，将 X10 ～ X1B 的 BCD 数据进行 BIN 转换后，存储到 Var_D8 中的程序。



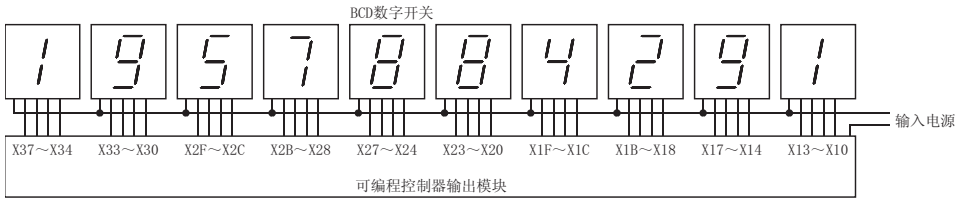
[结构体梯形图]



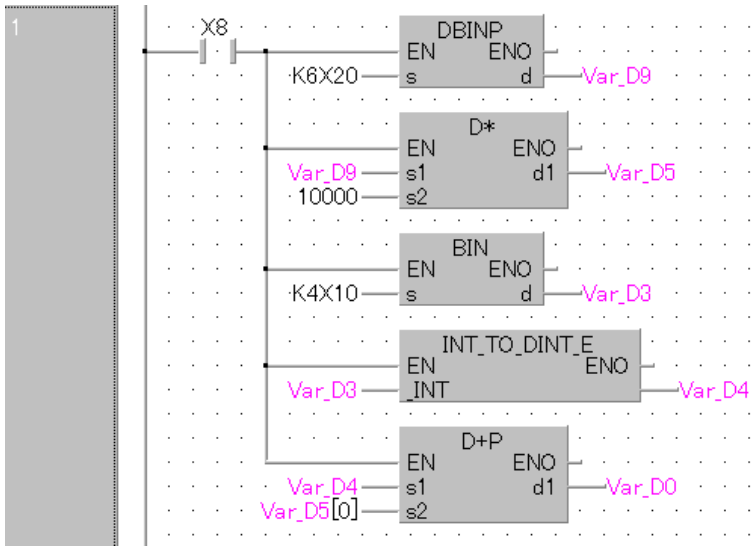
[ST]

```
BINP(X8,K3X10,Var_D8);
```

(2) 以下为 X8 变为 ON 时，将 X10 ～ X37 的 BCD 数据进行 BIN 转换后，存储到 Var_D0 中的程序。
（是将 X20 ～ X37 的 BCD 数据进行 BIN 转换及将 X10 ～ X1F 的 BCD 数据进行 BIN 转换后的加法运算）



[结构体梯形图]




```
[ST]
IF X8 THEN
     DBINP(TRUE,K6X20,Var_D9);
     Var_D5[0]:=Var_D9*10000;
     BIN(TRUE,K4X10,Var_D3);
     INT_TO_DINT_E(TRUE,Var_D3,Var_D4);
     Var_D0:=Var_D4+Var_D5[0];
END_IF;
```

X10 ～ X37 设置了超过 2147483647 的 BCD 值的情况下，由于超出了 32 位软元件的可处理数值范围，因此 Var_D0 的值将变为负值。

6.3.3 BIN16 位 /32 位数据→浮动小数点转换（单精度）

FLT, DFLT

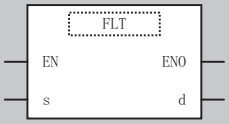
Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

FLT(P)
DFLT(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= FLT (EN, s, d);

中放入下述指令。

FLT	FLTP
DFLT	DFLTP

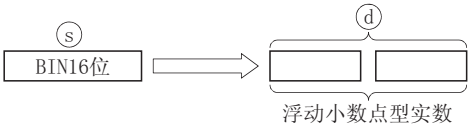
输入自变量，	EN:	执行条件	：位
	s:	要转换为浮动小数点数据的整数数据	：ANY16/32
输出自变量，	ENO:	执行结果	：位
	d:	转换后的浮动小数点数据	：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○	○		○		-
ⓓ	-	○		-	○		-		-

功 能

FLT(P)

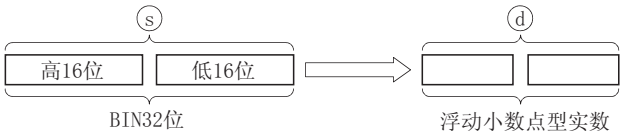
(1) 将ⓢ 中可指定的 BIN16 位数据转换为浮动小数点型实数后，将结果存储到ⓓ 中指定的软元件中。



(2) ⓢ 中指定的值为 BIN 值且在 -32768 ～ 32767 的范围内。

DFLT(P)

(1) 将ⓢ 中指定的 BIN32 位数据转换为浮动小数点型实数后，存储到ⓓ 中指定的软元件中。

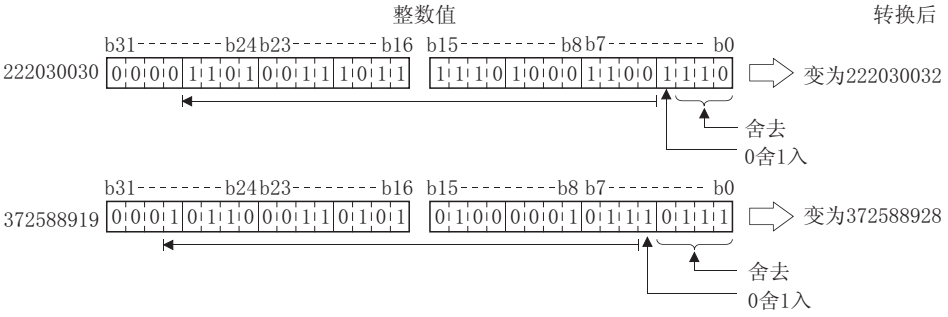


(2) ⓢ 中可指定的值为 BIN 值且在 -2147483648 ～ 2147483647 的范围内。

(3) 由于浮动小数点型实数是以 32 位的单精度进行处理，因此以 2 进制数表示时的有效位数为 24 位，以 10 进制数表示时约为 7 位。

因此，整数值超出了 -16777216 ～ 16777215(24 位 BIN 值）的范围的情况下，转换的值将产生误差。

对于转换结果，将整数值的高位算起的第 25 位进行 0 舍 1，将第 26 位及以后的值舍去。



出 错

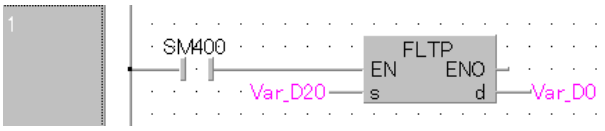
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出了以下范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 |　　（出错代码： 4141)

程序示例

(1) 以下为将 Var_D20 的 BIN16 位数据转换为浮动小数点型实数后，存储到 Var_D0 中的程序。

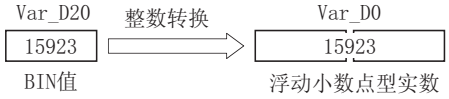
[结构体梯形图]



[ST]

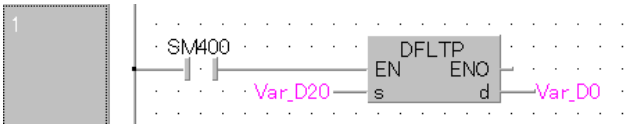
```
FLTP(SM400,Var_D20,Var_D0);
```

[动作]



(2) 以下为将 Var_D20 的 BIN32 位数据转换为浮动小数点型实数后，存储到 Var_D0 中的程序。

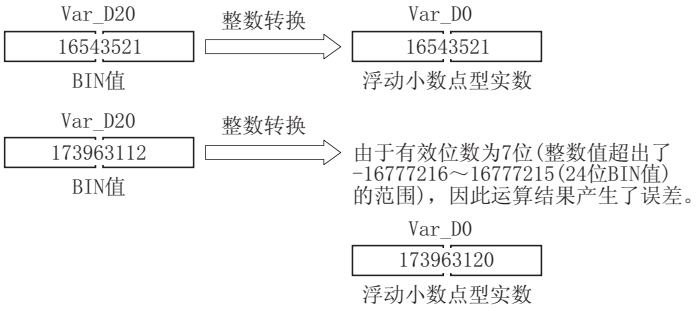
[结构体梯形图]



[ST]

```
DFLTP(SM400,Var_D20,Var_D0);
```

[动作]



6.3.4 BIN16 位 /32 位数据→浮动小数点转换（双精度）

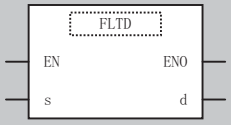
FLTD, DFLTD

Basic | High performance | Universal | L CPU

FLTD(P)
DFLTD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= FLTD (EN, s, d);

[　　] 中放入下述指令。

FLTD	FLTDP
DFLTD	DFLTDP

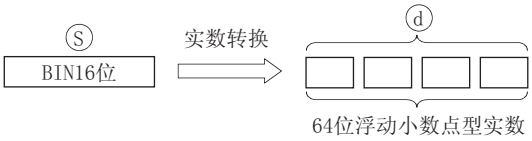
输入自变量，EN:　执行条件　：位
　　s:　要转换为 64 位浮动小数点型实数数据的整数数据　：ANY16/32
输出自变量，ENO:　执行结果　：位
　　d:　转换后的 64 位浮动小数点型实数数据　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-			○		-
ⓓ	-	○		-			-		-

功 能

FLTD(P)

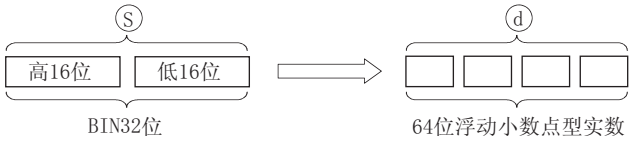
(1) 将ⓢ 中指定的 BIN16 位数据转换为 64 位浮动小数点型实数数据后，存储到ⓓ 中指定的软元件中。



(2) ⓢ 中可指定的值为 BIN 值且在 -32768 ～ 32767 的范围内。

DFLTD(P)

(1) 将ⓢ 中指定的 BIN32 位数据转换为 64 位浮动小数点型实数数据后，将结果存储到ⓓ 中。



(2) ⓢ +1, ⓢ 中可指定的值为 BIN 值且在 -2147483648 ～ 2147483647 的范围内。

出 错

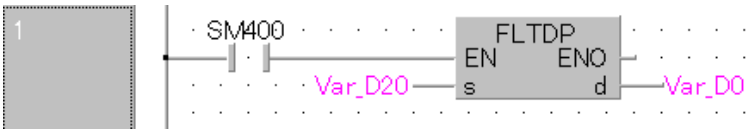
在以下情况下将发生运算出错，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- 运算结果超出了下述范围时。(发生了上溢时。)
  2^{128} ≦ | 运算结果 | (出错代码： 4141)

程序示例

(1) 以下为将 Var_D20 的 BIN16 位数据转换为 64 位浮动小数点型实数后，存储到 Var_D0 中的程序。

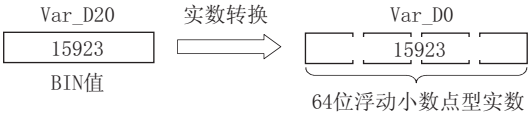
[结构体梯形图]



[ST]

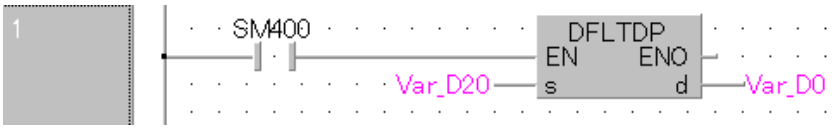
```
FLTDP(SM400,Var_D20,Var_D10);
```

[动作]



(2) 以下为将 Var_D20 的 BIN32 位数据转换为 64 位浮动小数点型实数后，存储到 Var_D0 中的程序。

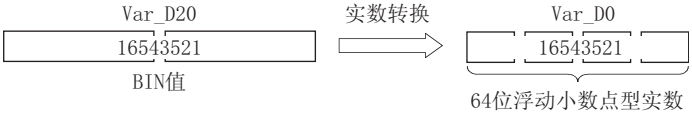
[结构体梯形图]



[ST]

```
DFLTDP(SM400,Var_D20,Var_D0);
```

[动作]



6.3.5 浮动小数点数据→BIN16 位 /32 位转换（单精度）

INT, DINT

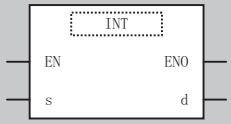
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122” 以后

INT(P)
DINT(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= INT (EN, s, d);

□ 中放入下述指令。

INT	INTP
DINT	DINTP

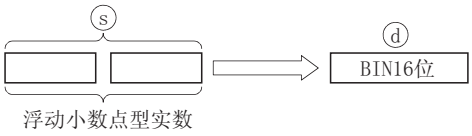
输入自变量，	EN:	执行条件	：位
	s:	要转换为 BIN 值的浮动小数点数据	：ANY32，单精度实数
输出自变量，	ENO:	执行结果	：位
	d:	转换后的 BIN 值	：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－		○	－	○	－
ⓓ	○	○		○		○	○	－	－

功 能

INT(P)

(1) 将ⓢ 中指定的浮动小数点型实数转换为 BIN16 位数据后，存储到ⓓ 中指定的软元件中。



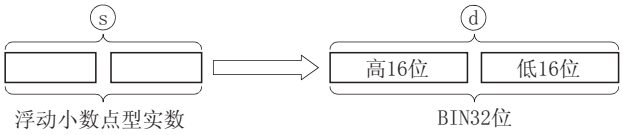
(2) 将ⓢ 中指定的浮动小数点型实数可在 -32768 ～ 32767 的范围内进行指定。

(3) ⓓ 中存储的整数值是以 BIN16 位进行存储。

(4) 对于转换后的数据，将实数的小数点以下的第 1 位进行四舍五入。

DINT(P)

(1) 将ⓢ 中指定的浮动小数点型实数转换为 BIN32 位数据后，存储到ⓓ 中指定的软元件中。



(2) 将ⓢ 中指定的浮动小数点型实数可在 -2147483648 ～ 2147483647 的范围内进行指定。

(3) ⓓ 中存储的整数值是以 BIN32 位进行存储。

(4) 对于转换后的数据，将实数的小数点以下第 1 位进行四舍五入。

出 错

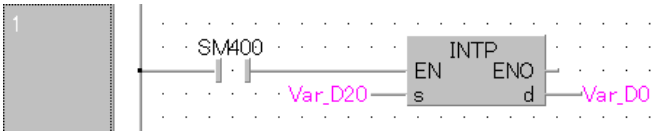
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。(通用型 QCPU、LCPU 时)
  0, 2^{-126} ≦ | 指定软元件的内容 | < 2^{128} (出错代码： 4140)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  (通用型 QCPU、LCPU 时) (出错代码： 4140)
- 使用 INT 指令时，ⓢ 中设置的 32 位浮动小数点型数据超出了 32768 ～ 32767 的范围时。
  (出错代码： 4100)
- 使用 DINT 指令时，ⓢ 中设置的 32 位浮动小数点型数据超出了 2147483648 ～ 2147483647 的范围时。 (出错代码： 4100)

程序示例

(1) 以下为将 Var_D20 的浮动小数点型实数转换为 BIN16 位数据后，存储到 Var_D0 中的程序。

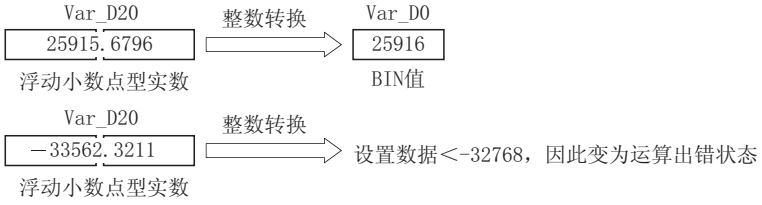
[结构体梯形图]



[ST]

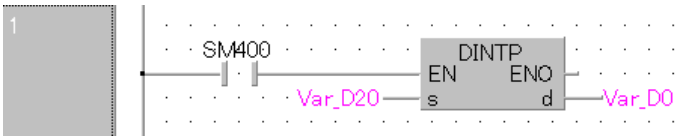
```
INTP(SM400,Var_D20,Var_D0);
```

[动作]



(2) 以下为将 Var_D20 的浮动小数点型实数转换为 BIN32 位数据后，存储到 Var_D0 中的程序。

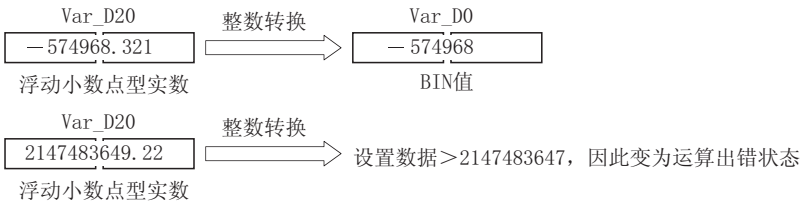
[结构体梯形图]



[ST]

```
DINTP(SM400,Var_D20,Var_D0);
```

[动作]



6.3.6 浮动小数点数据→BIN16 位 /32 位转换（双精度）

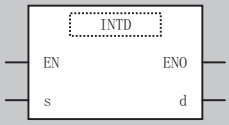
INTD, DINTD

Basic ✗ High performance ✗ Universal L CPU

INTD(P)
DINTD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= INTD (EN, s, d);

中放入下述指令。

INTD	INTDP
DINTD	DINTDP

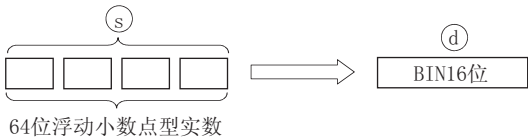
输入自变量，EN:　执行条件　：位
　　s:　要转换为 BIN 值的 64 位浮动小数点型实数数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　转换后的 BIN 值　：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－			－	○	－
ⓓ	－	○		－			○	－	－

功 能

INTD(P)

(1) 将ⓢ 中指定的 64 位浮动小数点型实数数据转换为 BIN16 位数据后，存储到ⓓ 中指定的软元件中。



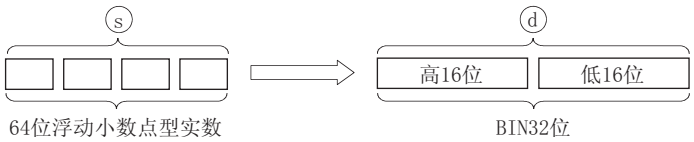
(2) ⓢ +3, ⓢ +2, ⓢ +1, ⓢ 中可指定的 64 位浮动小数点型实数数据的范围为 -32768.0 ～ 32767.0。

(3) ⓓ 中存储的整数值是以 BIN16 位进行存储。

(4) 对于转换后的数据，将 64 位浮动小数点型实数数据的小数点以下第 1 位进行四舍五入。

DINTD(P)

(1) 将ⓢ中指定的 64 位浮动小数点型实数数据转换为 BIN32 位数据后，存储到ⓓ中指定的软元件中。



(2) ⓢ+3, ⓢ+2, ⓢ+1, ⓢ中可指定的 64 位浮动小数点型实数数据的范围为 -2147483648.0 ～ 2147483647.0。

(3) ⓓ+1,ⓓ中存储的整数值是以 BIN32 位进行存储。

(4) 对于转换后的数据，将 64 位浮动小数点型实数数据的小数点以下第 1 位进行四舍五入。

出 错

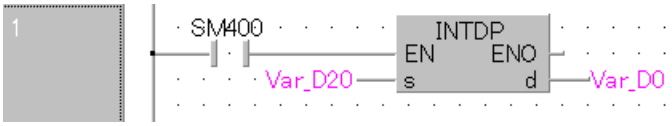
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 （出错代码： 4140)
  0, $2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码： 4140)
- 使用 INTD 指令使用时，ⓢ中设置的 64 位浮动小数点型数据超出了 -32768.0 ～ 32767.0 的范围时。 （出错代码： 4140)
- 使用 DINTD 指令时，ⓢ中设置的 64 位浮动小数点型数据超出了 -2147483648.0 ～ 2147483647.0 的范围时。 （出错代码： 4140)

程序示例

(1) 以下为将 Var_D20 的 64 位浮动小数点型实数转换为 BIN16 位数据后，存储到 Var_D0 中的程序。

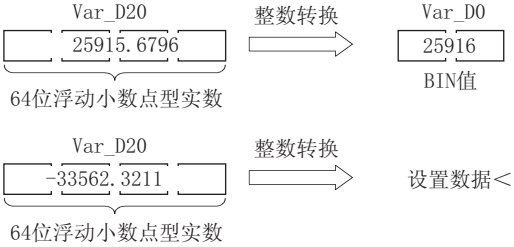
[结构体梯形图]



[ST]

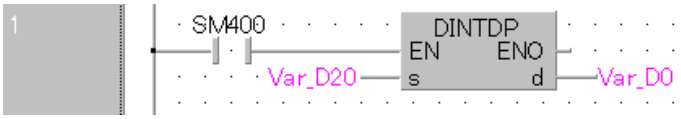
```
INTDP(SM400,Var_D20,Var_D0);
```

[动作]



(2) 以下为将 Var_D20 的 64 位浮动小数点型实数转换为 BIN32 位数据后，存储到 Var_D0 中的程序。

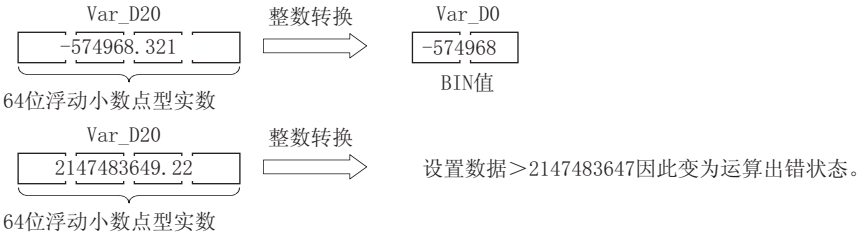
[结构体梯形图]



[ST]

```
DINTDP(SM400,Var_D20,Var_D0);
```

[动作]



6.3.7 BIN16 位数据→ BIN32 位数据转换

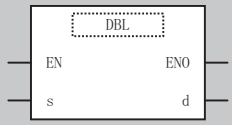
DBL

Basic | High performance | Universal | L CPU

DBL(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= DBL (EN, s, d);

中放入下述指令。

DBL　　DBLP

输入自变量，EN:　执行条件　：位

s:　BIN16 位数据或存储 BIN16 位数据的软元件的起始编号　：ANY16

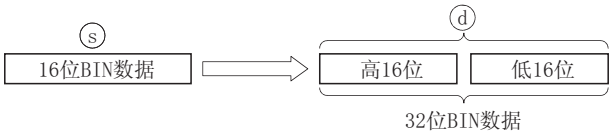
输出自变量，ENO:　执行结果　：位

d:　转换后的 BIN32 位数据　：ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○				○	-
ⓓ				○				-	-

功 能

将ⓢ 中指定的软元件的 BIN16 位数据转换为带符号 BIN32 位数据后，存储到ⓓ 中指定的软元件中。



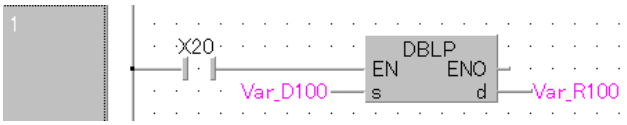
出 错

不存在 DBL(P) 指令相关的出错。

程序示例

以下为 X20 变为 ON 时，将 Var_D100 的 BIN16 位数据转换为 BIN32 位数据后，存储到 Var_R100 中的程序。

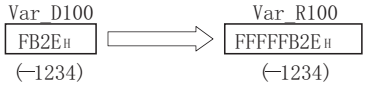
[结构体梯形图]



[ST]

```
DBLP(X20,Var_D100,Var_R100);
```

[动作]



6.3.8 BIN32 位数据→ BIN16 位数据转换

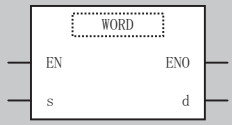
WORD

Basic | High performance | Universal | L CPU

WORD(P)

P: 执行条件 : ↓

结构体梯形图



ST

```
ENO:= WORD (EN, s, d);
```

□ 中放入下述指令。

WORD WORDP

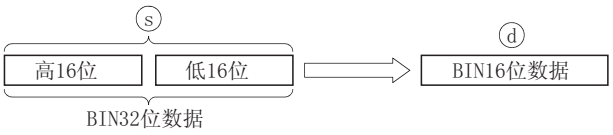
输入自变量，	EN:	执行条件	: 位
	s:	BIN32 位	: ANY32
输出自变量，	ENO:	执行结果	: 位
	d:	转换后的 BIN16 位数据	: ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ		○						○	-
ⓓ		○						-	-

功 能

将ⓢ中指定的软元件的 BIN32 位数据转换为带符号 BIN16 位数据后，存储到ⓓ中指定的软元件中。

可指定的范围为 -32768 ～ 32767。



出 错

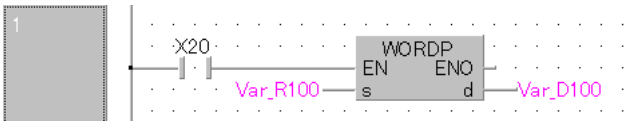
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的软元件的内容超出了 -32768 ～ 32767 的范围时。 （出错代码： 4100）

程序示例

以下为 X20 变为 ON 时，将 Var_R100 的 BIN32 位数据转换为 BIN16 位数据后，存储到 Var_D100 中的程序。

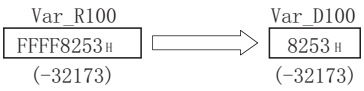
[结构体梯形图]



[ST]

```
WORDP(X20,Var_R100,Var_D100);
```

[动作]



6.3.9 BIN16 位 /32 位数据→格雷码转换

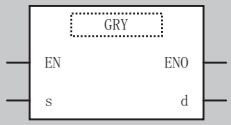
GRY, DGRY

Basic | High performance | Universal | L CPU

GRY(P)
DGRY(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= GRY (EN, s, d);

中放入下述指令。

GRY	GRYP
DGRY	DGRYP

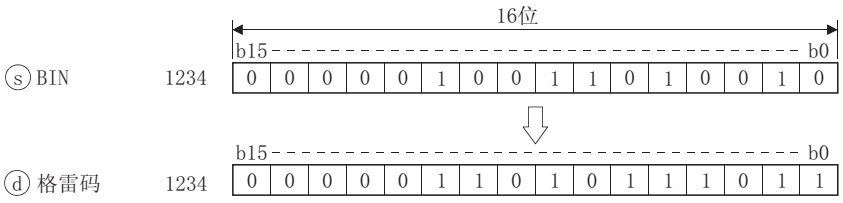
输入自变量， EN: 执行条件 : 位
s: BIN : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 转换后的格雷码 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ		○						○	-
ⓓ		○						-	-

功 能

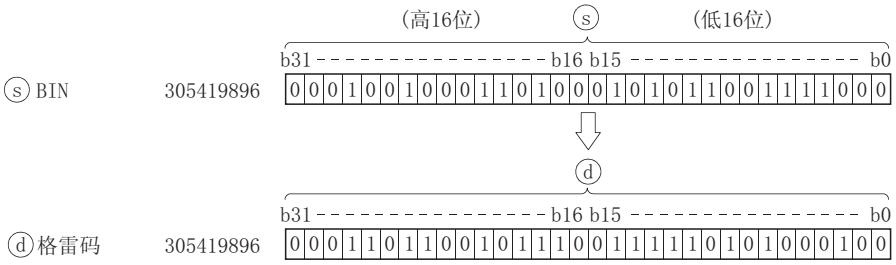
GRY(P)

将ⓢ中指定的软元件的 BIN16 位数据转换为格雷码后，存储到ⓓ中指定的软元件中。
ⓢ中不能指定负数。



DGRY(P)

将ⓢ中指定的软元件的 BIN32 位数据转换为格雷码后，存储到ⓓ中指定的软元件中。
ⓢ中不能指定负数。



出 错

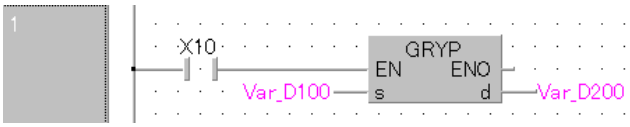
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ中的数据为负数时。 （出错代码： 4100）

程序示例

(1) 以下为 X10 变为 ON 时，将 Var_D100 的 BIN 数据转换为格雷码后，存储到 Var_D200 中的程序。

[结构体梯形图]

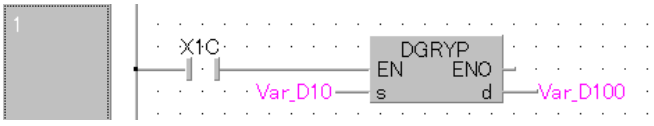


[ST]

```
GRYP(X10,Var_D100,Var_D200);
```

(2) 以下为 X1C 变为 ON 时，将 Var_D10 的 BIN 数据转换为格雷码后，存储到 Var_D100 中的程序。

[结构体梯形图]



[ST]

```
DGRYP(X1C,Var_D10,Var_D100);
```

6.3.10 格雷码→ BIN16 位 /32 位转换

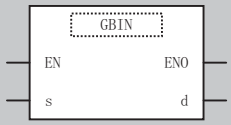
GBIN, DGBIN

Basic | High performance | Universal | L CPU

GBIN(P)
DGBIN(P)

P: 执行条件　　: ↓

结构体梯形图



ST

ENO:= GBIN (EN, s, d);

□ 中放入下述指令。

GBIN	GBINP
DGBIN	DGBINP

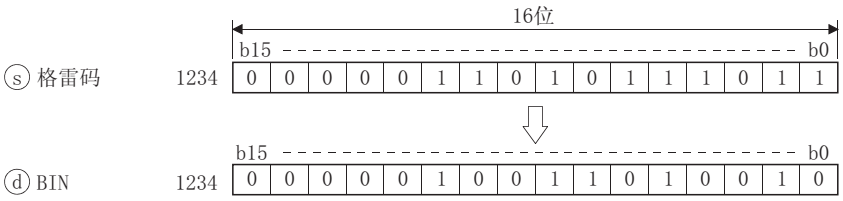
输入自变量， EN: 执行条件 : 位
s: 格雷码数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 转换后的 BIN 数据 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ		○						○	-
ⓓ		○						-	-

功 能

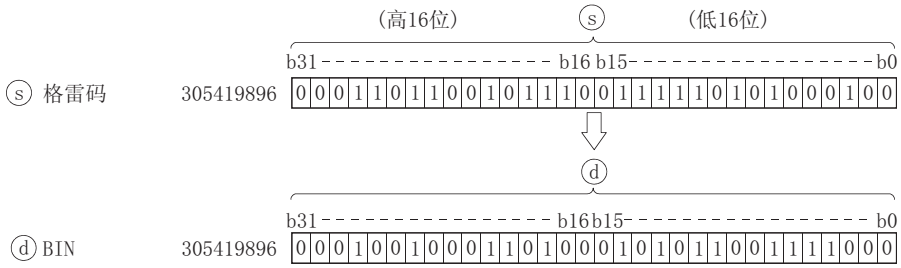
GBIN(P)

将ⓢ中指定的软元件中存储的格雷码数据转换为 BIN16 位数据后，存储到ⓓ中指定的软元件中。
ⓢ中可指定的范围为 0 ～ 32767。



DGBIN(P)

将ⓢ中指定的软元件中存储的格雷码数据转换为 BIN32 位数据后，存储到ⓓ中指定的软元件中。
ⓢ中可指定的范围为 0 ～ 2147483647。



出 错

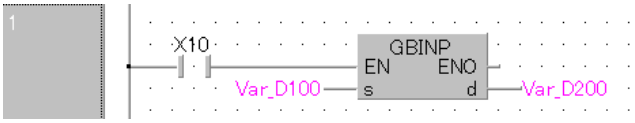
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- GBIN(P) 指令时，，ⓢ的数据超出了 0 ～ 32767 的范围时。　　(出错代码： 4100)
- DGBIN(P) 指令时，ⓢ的数据超出了 0 ～ 2147483647 的范围时。　(出错代码： 4100)

程序示例

(1) 以下为 X10 变为 ON 时，将 Var_D100 的格雷码数据转换为 BIN 数据后，存储到 Var_D200 中的程序。

[结构体梯形图]

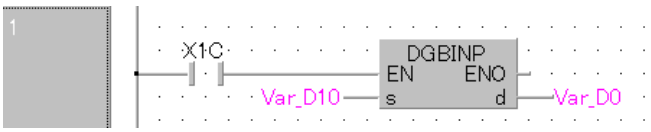


[ST]

```
GBINP(X10,Var_D100,Var_D200);
```

(2) 以下为 X1C 变为 ON 时，将 Var_D10 的格雷码数据转换为 BIN 数据后，存储到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
DGBINP(X1C,Var_D10,Var_D0);
```

6.3.11 BIN16 位 /32 位数据 2 的补数（符号取反）

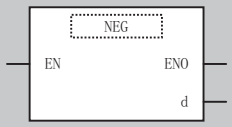
NEG, DNEG

Basic | High performance | Universal | L CPU

NEG(P)
DNEG(P)

（P：执行条件　　：↓）

结构体梯形图



ST

ENO:= NEG (EN, d);

□ 中放入下述指令。

NEG	NEGP
DNEG	DNEGP

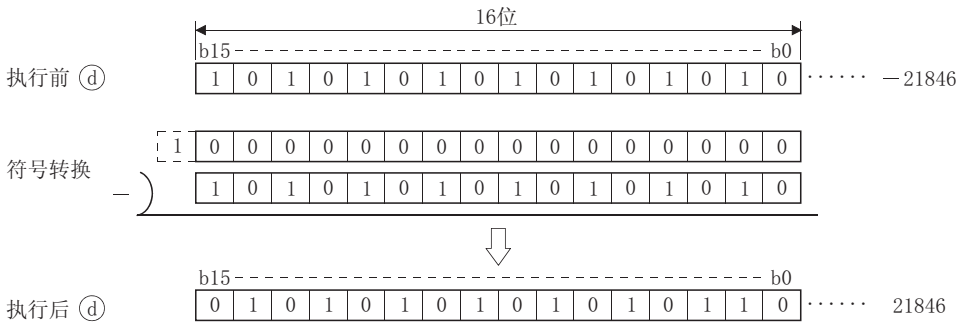
输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位
d:　进行 2 的补数的数据　：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	○							-	

功 能

NEG(P)

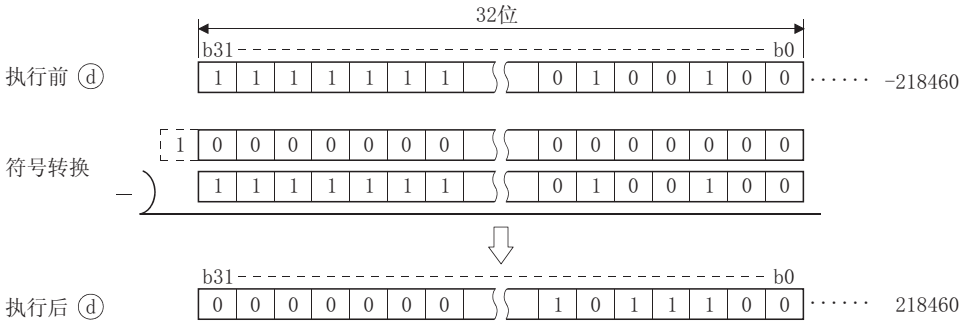
(1) 将ⓓ中指定的 16 位软元件的符号进行取反后，存储到ⓓ中指定的软元件中。



(2) 用于对正负符号进行取反。

DNEG(P)

(1) 将ⓓ中指定的 32 位软元件的符号进行取反后，存储到ⓓ中指定的软元件中。



(2) 用于对正负符号进行取反。

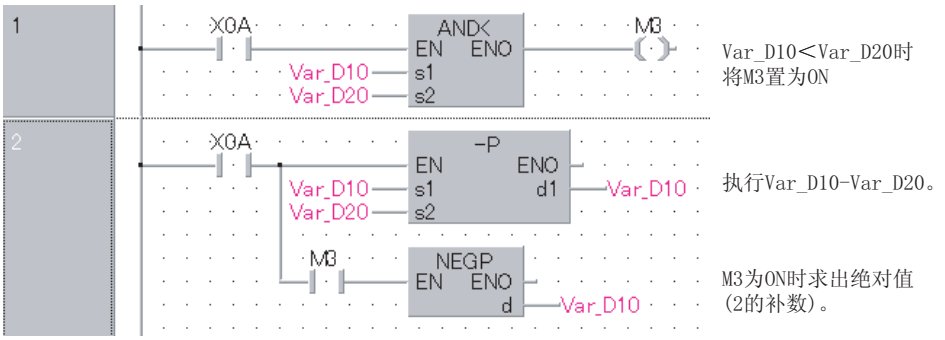
出 错

不存在 NEG(P)/DNEG(P) 指令的相关出错。

程序示例

以下为 X0A 为 ON 时进行 Var_D10-Var_D20 的计算，其结果为负时求出绝对值的程序。

[结构体梯形图]




```
[ST]
IF X0A THEN
    OUT(Var_D10<Var_D20,M3);
    Var_D10:=Var_D10-Var_D20;
    NEGP(M3,Var_D10);
END_IF;
```

6.3.12 浮动小数点符号取反（单精度）

ENEG

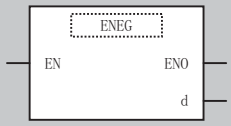
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

ENEG(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ENEG (EN, d);

[ ] 中放入下述指令。

ENEG　　ENEGP

输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位
d:　进行符号取反的浮动小数点数据　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	-	○		-	○		-		

功 能

(1) 将ⓓ中指定的软元件的浮动小数点型实数数据的符号进行取反后，存储到ⓓ中指定的软元件中。

(2) 用于对正负符号进行取反。

出 错

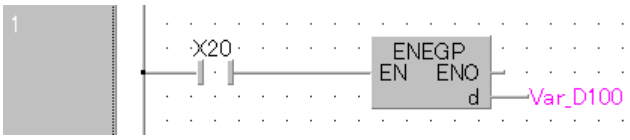
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（通用型 QCPU、LCPU 时）
  0, $2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$　（出错代码：4140）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）　（出错代码：4140）

程序示例

以下为 X20 变为 ON 时，将 Var_D100 的浮动小数点型实数数据的符号进行取反后，存储到 Var_D100 中的程序。

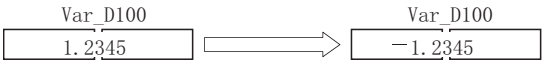
[结构体梯形图]



[ST]

```
ENEGP(X20,Var_D100);
```

[动作]



6.3.13 浮动小数点符号取反（双精度）

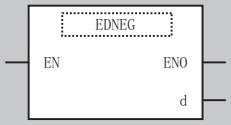
EDNEG

Basic　High performance　Universal　L CPU

EDNEG(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EDNEG (EN, d);

[　　] 中放入下述指令。

EDNEG　　EDNEGP

输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位
d:　进行符号取反的数据　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	-	○		-					

功 能

(1) 将ⓓ中指定的软元件的 64 位浮动小数点型实数数据的符号进行取反后，存储到ⓓ中指定的软元件中。
可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。

(2) 用于对正负符号进行取反。

出 错

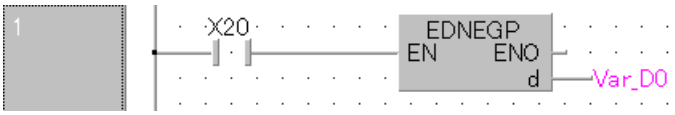
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。　（出错代码： 4140）
  0, $2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。　（出错代码： 4140）

程序示例

以下为 X20 变为 ON 时，将 Var_D0 的 64 位浮动小数点型实数的符号进行取反后，存储到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
EDNEGP(X20,Var_D0);
```

[动作]

6.3.14 块 BIN16 位数据→块 BCD4 位转换

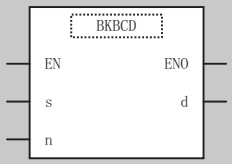
BKBCD

Basic | High performance | Universal | L CPU

BKBCD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BKBCD (EN, s, n, d);

中放入下述指令。

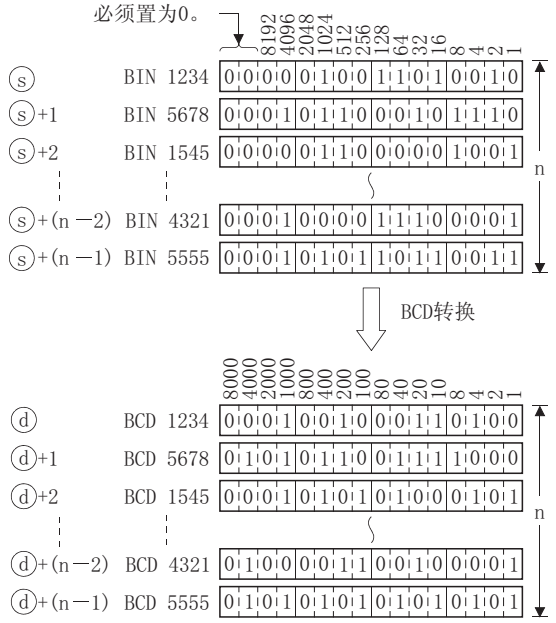
BKBCD　　BKBCDP

输入自变量，EN:　执行条件　：位
　　s:　存储 BIN 数据的软元件的起始编号　：ANY16
　　n:　转换数据数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　存储转换后的 BCD 数据的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	–	○		–					–
n	○	○		○					–
ⓓ	–	○		–					–

功 能

将ⓢ中指定的软元件算起 n 点的 BIN 数据（0 ～ 9999）进行 BCD 转换后，存储到ⓓ中指定的软元件以后。



出 错

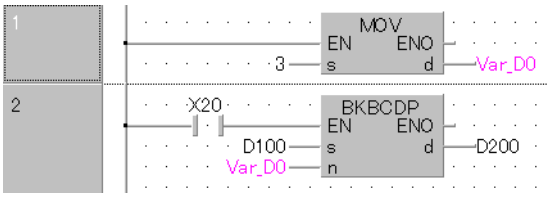
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ，ⓓ的软元件算起 n 点的范围超出了相应软元件时。（出错代码： 4101）
- ⓢ的软元件算起 n 点的数据超出了 0 ～ 9999 的范围时。（出错代码： 4100）
- ⓢ，ⓓ的软元件重叠时。（出错代码： 4101）

程序示例

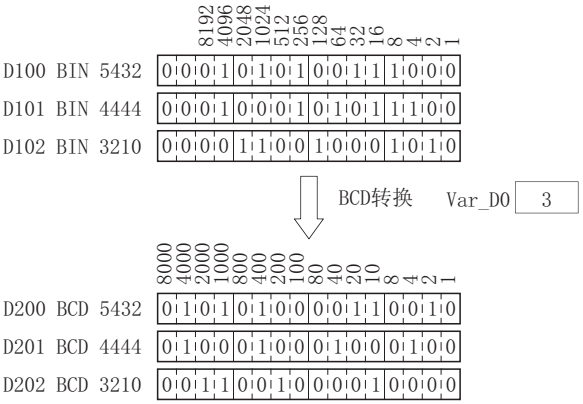
以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的 BIN 数据进行 BCD 转换后，将其结果存储到 D200 以后的程序。

[结构体梯形图]




```
[ST]
Var_D0:=3;
BKBCDP(X20,D100,Var_D0,D200);
```

[动作]



6.3.15 块 BCD4 位数据→块 BIN16 位转换

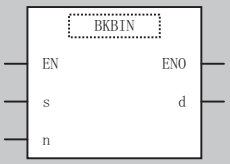
BKBIN

Basic | High performance | Universal | L CPU

BKBIN(P)

P: 执行条件　　: ↑

结构体梯形图



ST

ENO:= BKBIN (EN, s, n, d);

□ 中放入下述指令。

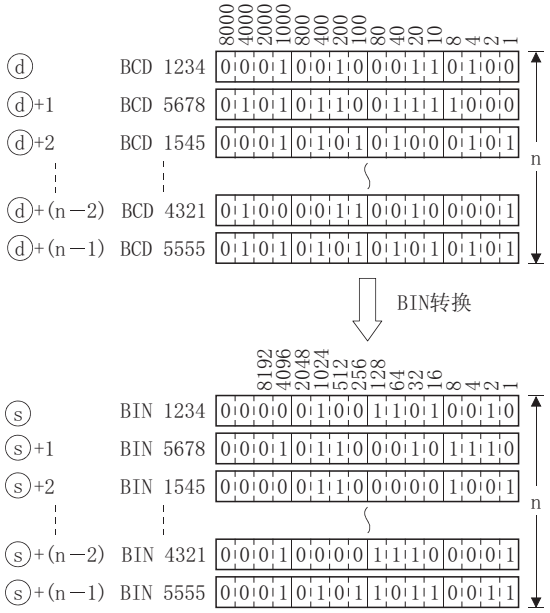
BKBIN	BKBINP

输入自变量，EN:　执行条件　：位
　　s:　存储 BCD 数据的软元件的起始编号　：ANY16
　　n:　变量数据数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　存储转换后的 BIN 数据的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	–	○		–					–
n	○	○		○					–
ⓓ	–	○		–					–

功 能

将ⓢ中指定的软元件算起 n 点的 BCD 数据 (0 ～ 9999) 进行 BIN 转换后，存储到ⓓ中指定的软元件以后。



出 错

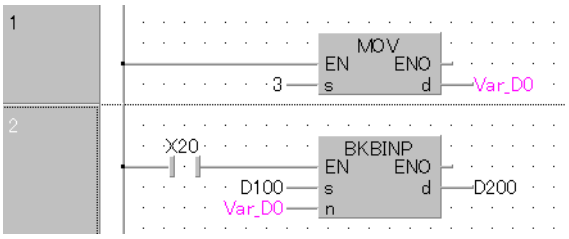
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ，ⓓ的软元件算起 n 点的范围超出了相应软元件时。 (出错代码：4101)
- ⓢ的软元件算起 n 点的数据超出了 0 ～ 9999 的范围时。 (出错代码：4100)
- ⓢ，ⓓ的软元件重叠时。 (出错代码：4101)

程序示例

以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的 BCD 数据进行 BIN 转换后，将其结果存储到 D200 以后的程序。

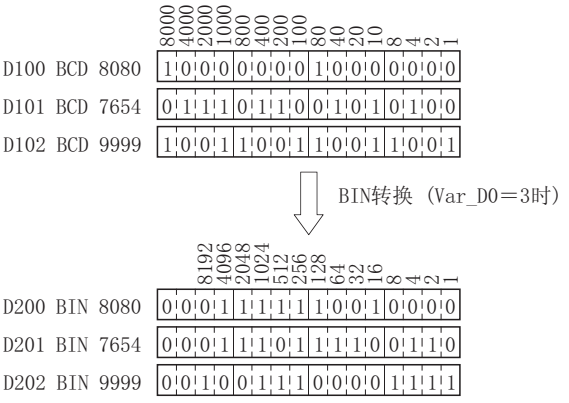
[结构体梯形图]



[ST]

```
Var_D0:=3;
BKBINP(X20,D100,Var_D0,D200);
```

[动作]



6.3.16 单精度→双精度转换指令

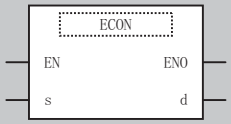
ECON

Basic　High performance　Universal　L CPU

ECON(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ECON (EN, s, d);

[　　] 中放入下述指令。

ECON　　ECONP

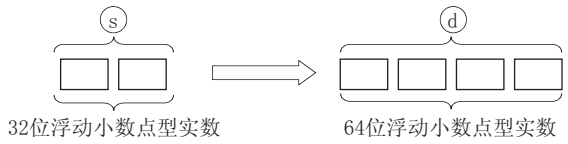
输入自变量，EN:　执行条件　：位
　　s:　转换源的数据　：单精度实数
输出自变量，ENO:　执行结果　：位
　　d:　转换后的数据　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

将ⓢ中指定的 32 位浮动小数点型实数数据转换为 64 位浮动小数点型实数数据后，将转换结果存储到ⓓ中。

ⓢ中可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。



出 错

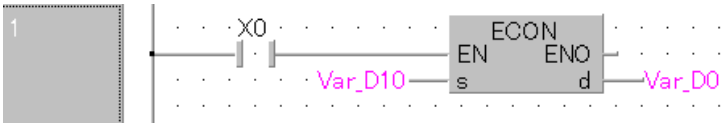
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 转换结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$转换结果$|$（出错代码： 4141）

程序示例

以下 X0 变为 ON 时，将 Var_D10 的 32 位浮动小数点型实数转换为 64 位浮动小数点型实数后，输出到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
ECON(X0,Var_D10,Var_D0);
```

6.3.17 双精度→单精度转换指令

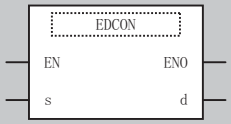
EDCON

Basic　High performance　Universal　L CPU

EDCON(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EDCON (EN, s, d);

[　　] 中放入下述指令。

EDCON　　EDCONP

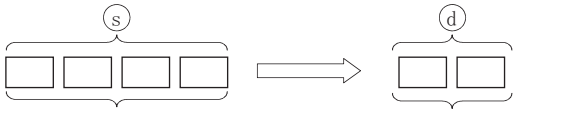
输入自变量，EN:　执行条件　：位
　　s:　转换源的数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　转换后的数据　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	-	○		-			-	○	-
ⓓ	-	○		-			○	-	

功 能

将ⓢ中指定的64位浮动小数点型实数数据转换为32位浮动小数点型实数数据后，将转换结果存储到ⓓ中指定的软元件中。

ⓢ中可指定的范围为0或 $2^{-1022} \sim 2^{1024}$。



出 错

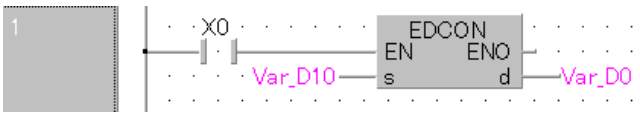
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 转换结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$转换结果$|$（出错代码： 4141）

程序示例

以下为 X0 变为 ON 时，将 Var_D10 的 64 位浮动小数点型实数转换为 32 位浮动小数点型实数后，输出到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
EDCON(X0,Var_D10,Var_D0);
```

6.4 数据传送指令

6.4.1 16 位 /32 位数据传送

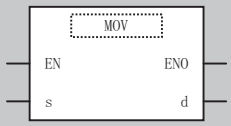
MOV, DMOV

Basic | High performance | Universal | L CPU

MOV(P)
DMOV(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= MOV (EN, s, d);

中放入下述指令。

MOV	MOVP
DMOV	DMOVP

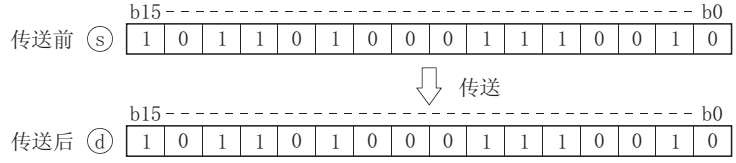
输入自变量， EN: 执行条件 : 位
s: 传送源的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 传送目标的数据 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○							○	-
ⓓ	○							-	-

功 能

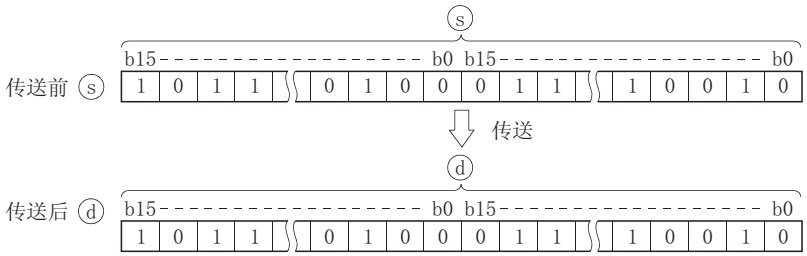
MOV(P)

将ⓢ中指定的软元件的 16 位数据，传送至ⓓ中指定的软元件中。



DMOV(P)

将ⓢ中指定的软元件的 32 位数据，传送至ⓓ中指定的软元件中。



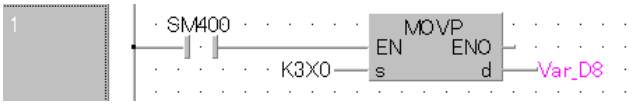
出 错

不存在 MOV(P)、DMOV(P) 指令相关的运算出错。

程序示例

(1) 以下为将输入 X0 ～ XB 的数据存储到 Var_D8 中的程序。

[结构体梯形图]

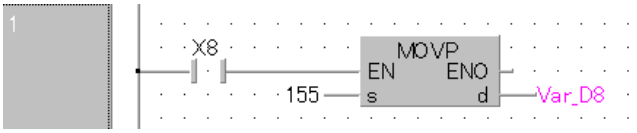


[ST]

```
MOVP(SM400,K3X0,Var_D8);
```

(2) 以下为 X8 变为 ON 时，将 155 传送至 Var_D8 中的程序。

[结构体梯形图]

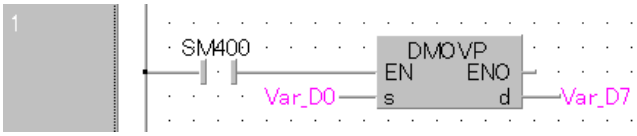


[ST]

```
MOVP(X8,155,Var_D8);
```

(3) 以下为将 Var_D0 的数据存储到 Var_D7 中的程序。

[结构体梯形图]

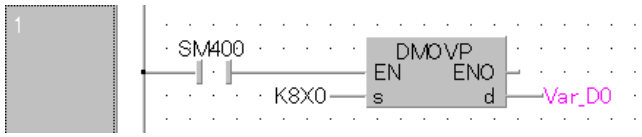


[ST]

```
DMOVP(SM400,Var_D0,Var_D7);
```

(4) 以下为将 X0 ～ X1F 的数据存储到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
DMOVP(SM400,K8X0,Var_D0);
```

6.4.2 浮动小数点数据传送（单精度）

EMOV(P)

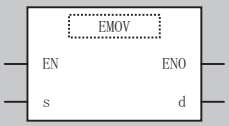
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

EMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EMOV (EN, s, d);

[　　] 中放入下述指令。

EMOV　　EMOVP

输入自变量，EN:　执行条件　：位
　　s:　要传送的数据　：单精度实数
输出自变量，ENO:　执行结果　：位
　　d:　传送目标的数据　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－	○		－	○	－
ⓓ	－	○		－	○		－	－	－

功 能

将ⓢ中指定的软元件中存储的浮动小数点型实数数据，传送至ⓓ中指定的软元件中。

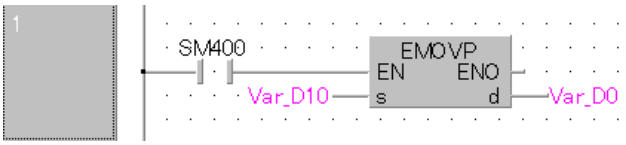
出 错

不存在 EMOV(P) 指令相关的运算出错。

程序示例

(1) 以下为将 Var_D10 的实数存储到 Var_D0 中的程序。

[结构体梯形图]



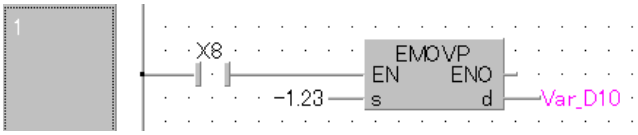
[ST]

```
EMOVP(SM400,Var_D10,Var_D0);
```

[动作]

(2) 以下为 X8 变为 ON 时，将实数 -1.23 存储到 Var_D10 中的程序。

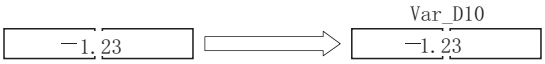
[结构体梯形图]



[ST]

```
EMOVP(X8,-1.23,Var_D10);
```

[动作]



6.4.3 浮动小数点数据传送（双精度）

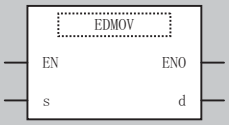
EDMOV

Basic　High performance　Universal　L CPU

EDMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EDMOV (EN, s, d);

□ 中放入下述指令。

EDMOV　　EDMOVP

输入自变量，EN:　执行条件　：位
　　s:　要传送的数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　传送目标的数据　：双精度实数

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

将ⓢ中指定的64位浮动小数点型实数数据传送至ⓓ中指定的软元件中。

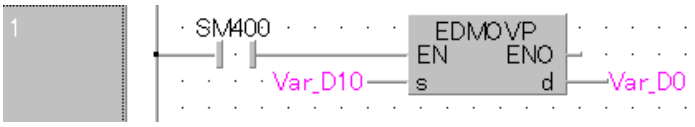
出 错

不存在EDMOV(P)指令相关的运算出错。

程序示例

(1) 以下为将 Var_D10 的 64 位浮动小数点型实数存储到 Var_D0 中的程序。

[结构体梯形图]



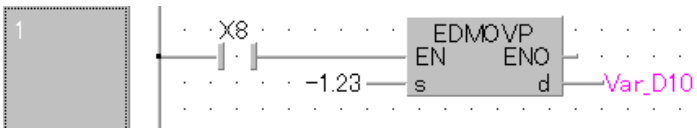
[ST]

```
EDMOVP(SM400,Var_D10,Var_D0);
```

[动作]

(2) 以下为 X8 变为 ON 时，将实数 -1.23 存储到 Var_D10 中的程序。

[结构体梯形图]



[ST]

```
EDMOVP(X8,-1.23,Var_D10);
```

[动作]

6.4.4 字符串传送

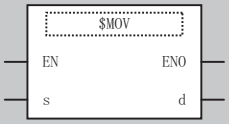
$MOV

Basic | High performance | Universal | L CPU

$MOV(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

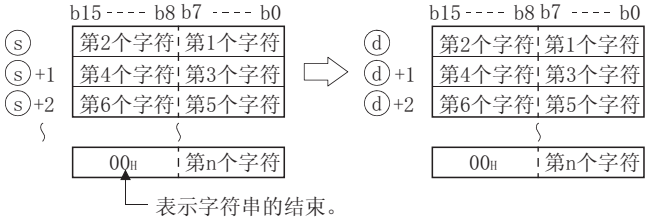
$MOV　　$MOVP

输入自变量，EN：　执行条件　：位
　　s：　传送字符串（最大字符串：32 字符）　：字符串
输出自变量，ENO：　执行结果　：位
　　d：　传送字符串　：字符串

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

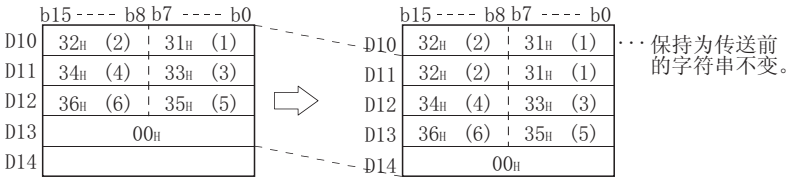
功 能

(1) 将ⓢ中指定的字符串数据，传送至ⓓ中指定的软元件编号以后。
在字符串的传送中，将ⓢ中指定的用 “ ”（双引号）围住的字符串或从软元件编号起至存储了 00H 的软元件编号为止的字符串进行一次传送。

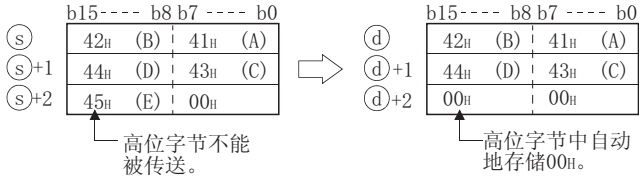


(2) 即使存储要传送的字符串数据的软元件范围（ⓢ～ⓢ+n）与存储传送的字符列数据的软元件范（ⓓ～ⓓ+n）重叠，也可正常处理。

将 D10 ～ D13 中存储的字符串传送至 D11 ～ D14 时的情况如下所示。



(3) ⓢ+n 的低位字节中存储了 00H 的情况下，在ⓓ+n 的高位字节、低位字节中均存储 00H。



出 错

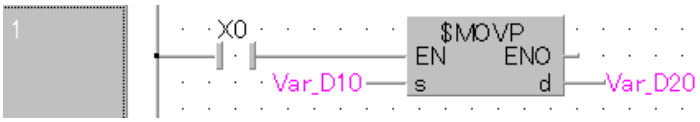
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的软元件编号以后，至相应软元件编号为止的区间不存在 00H 时。
  （出错代码： 4101）
- ⓓ中指定的软元件编号以后，至相应软元件的最终软元件编号为止的点数中，无法存储指定的全部字符串时。（出错代码： 4101）
- ⓢ的字符串超过了 16383 个字符时。（出错代码： 4101）

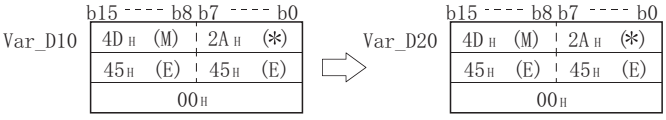
程序示例

(1) 以下为 X0 变为 ON 时，将 Var_D10 中存储的字符串数据，传送至 Var_D20 中的程序。

[结构体梯形图]

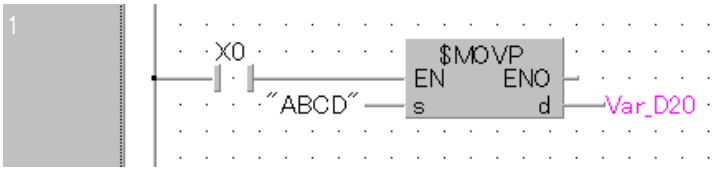


[动作]



(2) 以下为将 X0 置为 ON 时，将字符串 “ABCD”传送至 Var_D20 中的程序。

[结构体梯形图]



6.4.5 16 位 /32 位数据否定传送

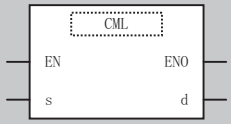
CML, DCML

Basic | High performance | Universal | L CPU

CML(P)
DCML(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= CML (EN, s, d);

中放入下述指令。

CML	CMLP
DCML	DCMLP

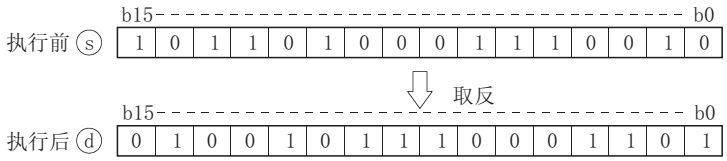
输入自变量， EN: 执行条件 : 位
s: 被取反的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 取反结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ		○						○	-
ⓓ		○						-	-

功 能

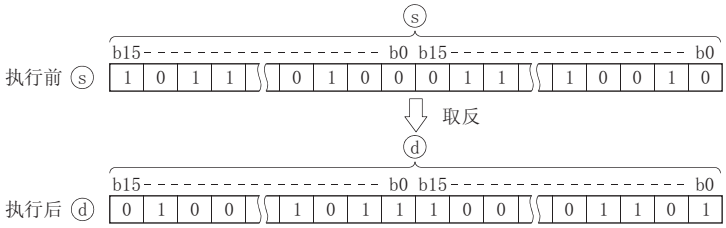
CML(P)

将ⓢ中指定的 16 位数据逐位进行取反后，将其结果传送至ⓓ中指定的软元件中。



DCML(P)

将ⓢ中指定的 32 位数据逐位进行取反后，将其结果传送至ⓓ中指定的软元件中。



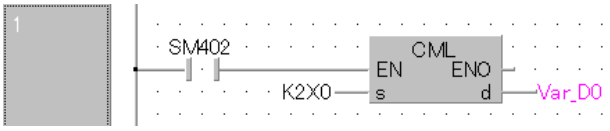
出 错

不存在 CML(P)、DCML(P) 指令相关的运算出错。

程序示例

(1) 以下为将 X0 ～ X7 的数据进行取反后，传送至 Var_D0 中的程序。

[结构体梯形图]

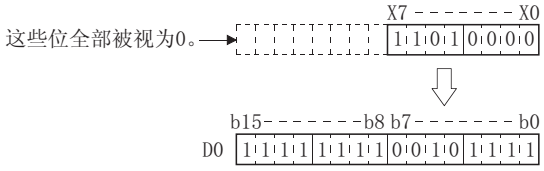


[ST]

```
CML(SM402,K2X0,Var_D0);
```

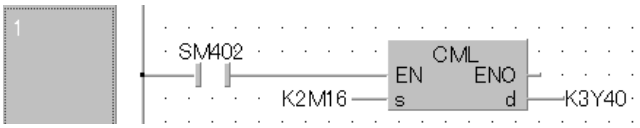
[动作]

ⓢ的位数 ＜ ⓓ的位数时



(2) 以下为将 M16 ～ M23 的数据进行取反后，传送至 Y40 ～ Y47 中的程序。

[结构体梯形图]

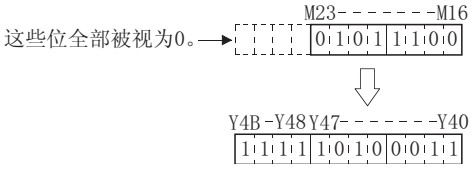


[ST]

```
CML(SM402,K2M16,K3Y40);
```

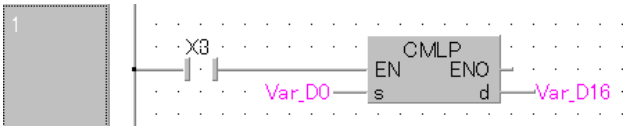
[动作]

ⓢ的位数 ＜ ⓓ的位数时



(3) 以下为 X3 变为 ON 时，将 Var_D0 的数据进行取反后，存储到 Var_D16 中的程序。

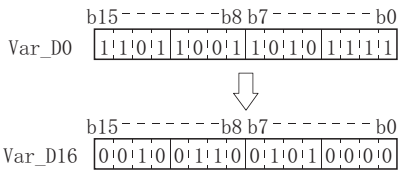
[结构体梯形图]



[ST]

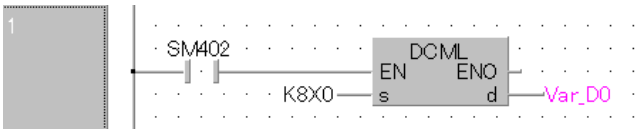
```
CMLP(X3,Var_D0,Var_D16);
```

[动作]



(4) 以下为将 X0 ～ X1F 的数据进行取反后，传送至 Var_D0 中的程序。

[结构体梯形图]

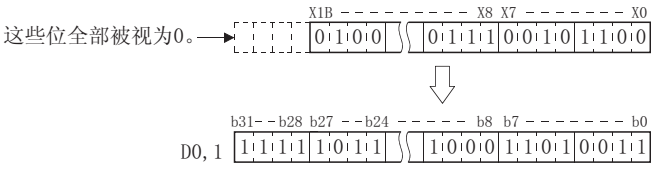


[ST]

```
DCML(SM402,K8X0,Var_D0);
```

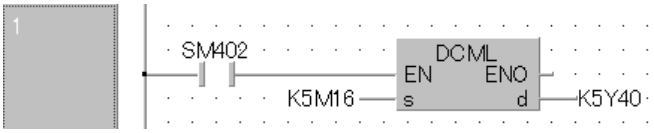
[动作]

ⓢ的位数 ＜ ⓓ的位数时



(5) 以下为将 M16 ～ M35 的数据进行取反后，传送至 Y40 ～ Y63 中的程序。

[结构体梯形图]

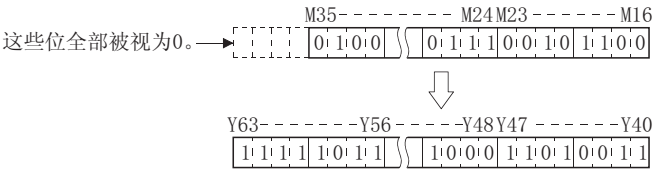


[ST]

```
DCML(SM402,K5M16,K6Y40);
```

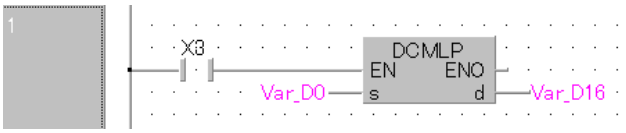
[动作]

ⓢ的位数 ＜ ⓓ的位数时



(6) 以下为 X3 变为 ON 时，将 Var_D0 的数据进行取反后，存储到 Var_D16 中的程序。

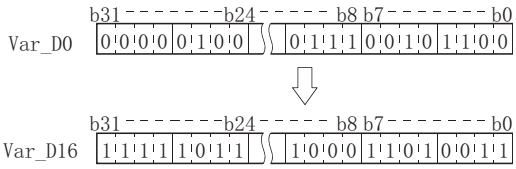
[结构体梯形图]



[ST]

```
DCMLP(X3,Var_D0,Var_D16);
```

[动作]



6.4.6 块 16 位数据传送

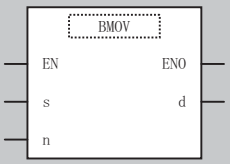
BMOV

Basic | High performance | Universal | L CPU

BMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BMOV (EN, s, n, d);

[　　] 中放入下述指令。

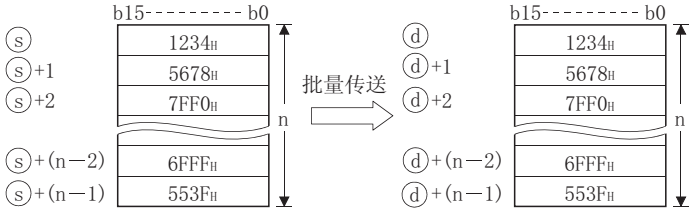
BMOV　　BMOVP

输入自变量，EN:　执行条件　：位
　　s:　存储要传送的数据的软元件的起始编号　：ANY16
　　n:　传送数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　传送目标的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ			○				-		-
n			○				○		-
ⓓ			○				-		-

功 能

(1) 将ⓢ中指定的软元件算起 n 点的 16 位数据，批量传送至ⓓ中指定的软元件算起的 n 点中。



(2) 即使传送源与传送目标的软元件重叠的情况下，也可以进行传送。
传送至软元件编号较小的方向时从ⓢ开始传送，传送至软元件编号较的方向时从ⓢ+(n-1) 开始传送。
但是，从 R 传送至 ZR 或从 ZR 传送至 R 的情况下，应注意按如下所示不要让 ZR 与 R 的各个传送范围重叠。
从 R 传送至 R 或从 ZR 传送至 ZR 的情况下，不存在问题。

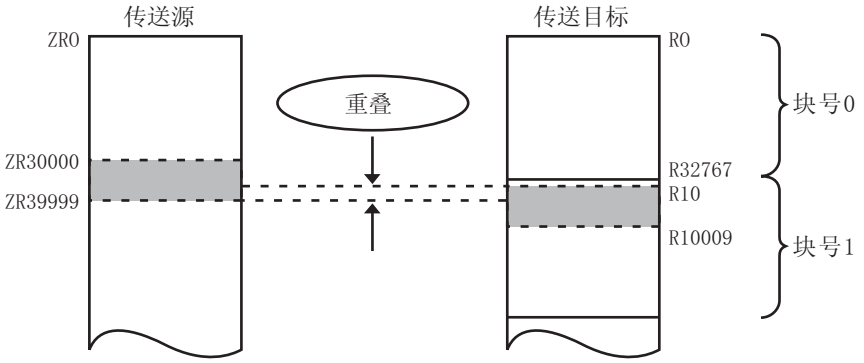
- ZR 的传送范围 ((指定的 ZR 起始 No.) ～ (指定的 ZR 起始 No.+ 传送数 -1))
- R 的传送范围 ((指定的 R 起始 No.+ 文件寄存器块号× 32768)
  ～ (指定的 R 起始 No.+ 文件寄存器块号× 32768+ 传送数 -1))

例

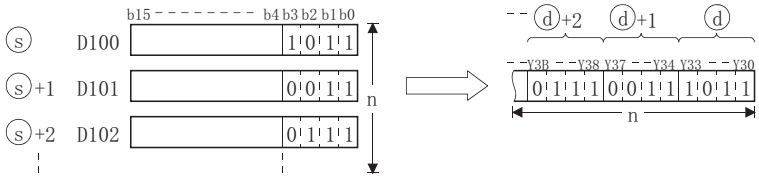
将 10000 点的数据从传送源 ZR30000 传送至传送目标的块号 1 的 R10 中的情况下，传送范围重叠。

- ZR 的传送范围 → (30000) ～ (30000+10000-1) → (30000) ～ (39999)
- R 的传送范围 → (10+(1×32768)) ～ (10+(1×32768)+10000-1)
  → (32778) ～ (42777)

因此，32778 至 39999 的范围重叠，值无法正常传送。



(3) ⓢ为字软元件而ⓓ为位软元件的情况下，位软元件的位数指定中指定的位数将成为字软元件的对象。
ⓓ中指定了 K1Y30 的情况下，ⓢ中指定的字软元件的低 4 位将成为对象。



(4) ⓢ, ⓓ中指定位软元件的情况下，必须将ⓢ, ⓓ的位数设置为相同。

(5) ⓢ, ⓓ中使用链接直接软元件及智能功能模块软元件的情况下，应只指定ⓢ或ⓓ中之一。

(6) 软元件范围检查的执行有无选择
通过软元件范围检查禁止标志 (SM237)，可以对执行 BMOV 指令时的软元件范围检查的执行有无进行选择。(仅子设置条件成立)
SM237 为 ON 时，不能对ⓢ～ⓢ(n-1)，ⓓ～ⓓ(n-1) 进行是否在软元件范围内的检查。

注意事项

SM237 为 ON 时不要进行下述访问。

- 变址修饰目标超出软元件范围的访问
- ⓓ～ ⓓ+(n-1) 跨越了软元件范围的边界的访问 *1
- 在未设置文件寄存器的状态下对文件寄存器进行的访问
- 对不存在多 CPU 间高速通信区域软元件的区域进行的访问
  (仅 QCPU(Q 模式))

*1: 请参阅 DFMOV 指令。

要 点

对于 SM237，只能使用下述 CPU 模块。

- 序列号的前 5 位数为 “10012”以后的通用型 QCPU
- LCPU

出 错

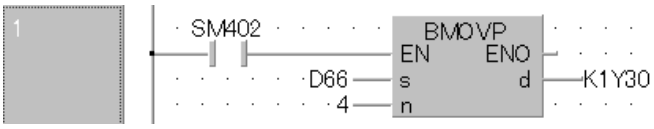
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ，ⓓ 算起的 n 点的软元件范围超出了相应软元件时。 (出错代码 : 4101)

程序示例

(1) 以下为将 D66 ～ D69 的低 4 位数据以 4 点为单位输出到 Y30 ～ Y3F 中的程序。

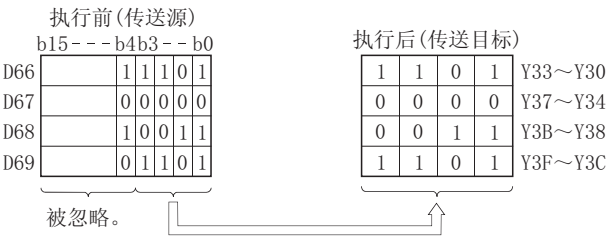
[结构体梯形图]



[ST]

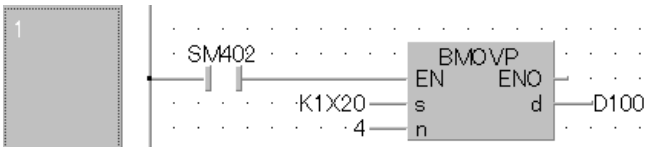
```
BMOVP(SM402,D66,4,K1Y30);
```

[动作]



(2) 以下为将 X20 ～ X2F 的数据以 4 点为单位输出到 D100 ～ D103 中的程序。

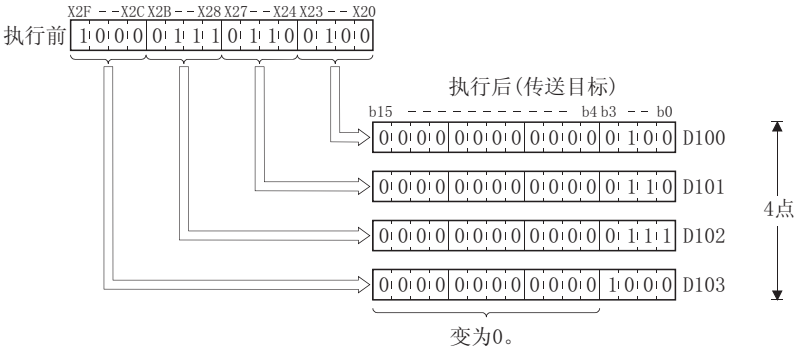
[结构体梯形图]



[ST]

```
BMOVP(SM402,K1X20,4,D100);
```

[动作]



6.4.7 同一 16 位数据块传送

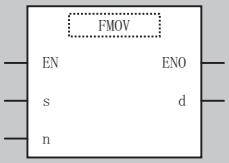
FMOV

Basic | High performance | Universal | L CPU

FMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= FMOV (EN, s, n, d);

[　　] 中放入下述指令。

FMOV　　FMOVP

输入自变量，EN:　执行条件　：位

s:　要传送的数据或存储要传送的数据的软元件的起始编号　：ANY16

n:　传送数　：ANY16

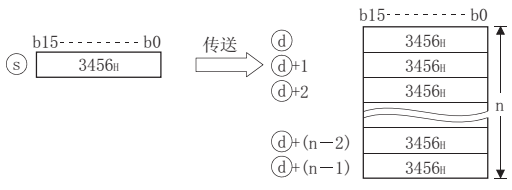
输出自变量，ENO:　执行结果　：位

d:　传送目标的软元件的起始编号　：ANY

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ			○				○		-
n			○				○		-
ⓓ			○				-		-

功 能

(1) 将ⓢ中指定的软元件的 16 位数据，传送至ⓓ中指定的软元件算起的 n 点中。



(2) ⓢ为字软元件而ⓓ为位软元件的情况下，位软元件的位数指定中指定的位数将成为ⓢ的字软元件的对象。

ⓓ中指定了 K1Y30 的情况下，ⓢ的字软元件的低 4 位将成为对象。

(3) ⓢ, ⓓ 中指定位软元件的情况下，必须将ⓢ, ⓓ 的位数设置为相同。

(4) 软元件范围检查的执行有无选择

通过软元件范围检查禁止标志 (SM237)，可以对执行 FMOV 指令时的软元件范围检查的执行有无进行选择。(仅子设置条件成立)

SM237 为 ON 时，不能对ⓢ～ⓢ (n-1), ⓓ～ⓓ (n-1) 进行是否在软元件范围内的检查。

注意事项

SM237 为 ON 时不要进行下述访问。

- 变址修饰目标超出软元件范围的访问
- ⓓ～ ⓓ +(n-1) 跨越了软元件范围的边界的访问 *1
- 在未设置文件寄存器的状态下对文件寄存器进行的访问
- 对不存在多 CPU 间高速通信区域软元件的区域进行的访问 (仅 QCPU(Q 模式))

*1: 请参阅 DFMOV 指令。

要 点

对于 SM237，只能使用下述 CPU 模块。

- 序列号的前 5 位数为 “10012” 以后的通用型 QCPU
- LCPU

出 错

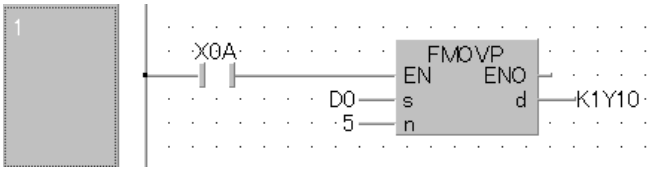
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 算起的 n 点的软元件范围超出了相应软元件时。 (出错代码 : 4101)

程序示例

(1) 以下为 X0A 变为 ON 时，将 D0 的低 4 位数据以 4 点为单位输出到 Y10 ～ Y23 中的程序。

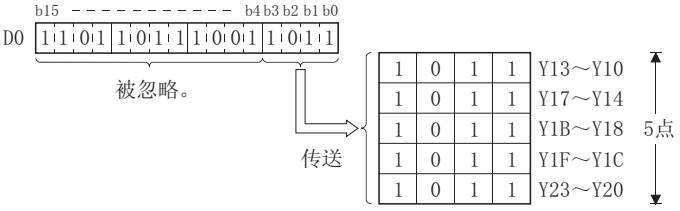
[结构体梯形图]



[ST]

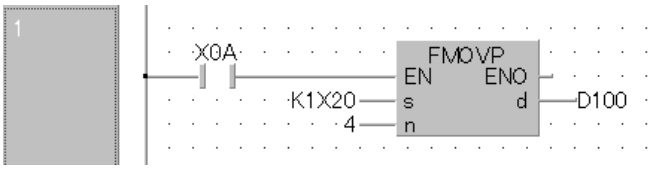
```
FMOVP(X0A,D0,5,K1Y10);
```

[动作]



(2) 以下为 X0A 变为 ON 时，将 X20 ～ X23 的数据输出到 D100 ～ D103 中的程序。

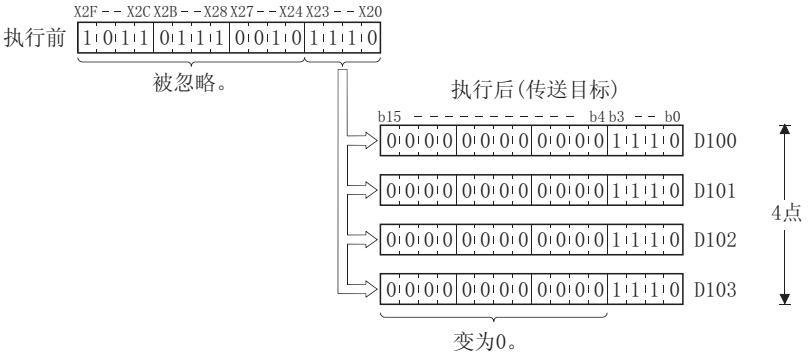
[结构体梯形图]



[ST]

```
FMOVP(X0A,K1X20,4,D100);
```

[动作]



6.4.8 同一 32 位数据块传送

DFMOV

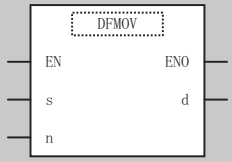
Basic | High performance | Ver. Universal | L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

DFMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= DFMOV (EN, s, n, d);

[ ] 中放入下述指令。

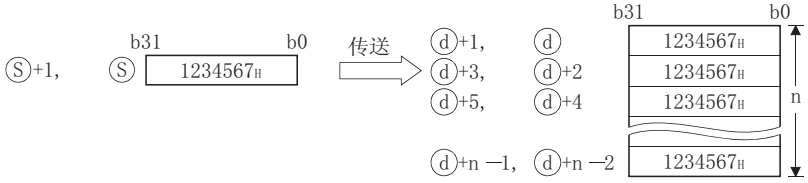
DFMOV　　DFMOVP

输入自变量，EN:　执行条件　：位
　　s:　要传送的数据或存储要传送的数据的软元件的起始编号　：ANY32
　　n:　传送数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　传送目标的软元件的起始编号　：ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○			○		-
n				○			○		-
ⓓ				○			-		-

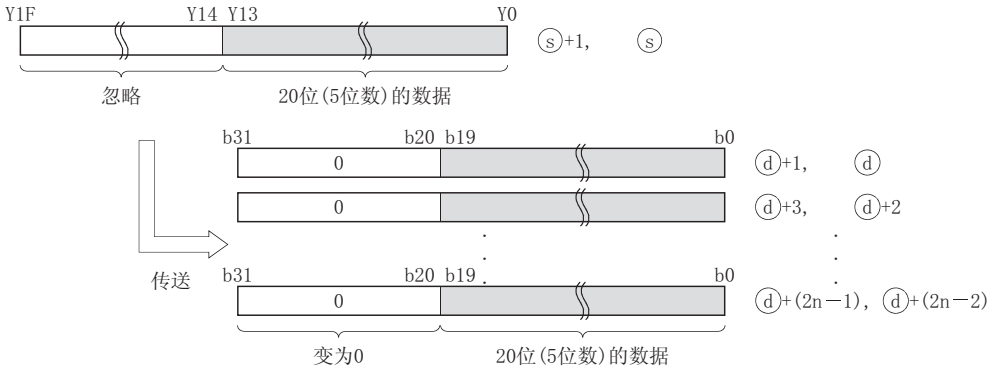
功 能

(1) 将ⓢ中指定的软元件的 32 位数据，传送至 ⓓ中指定的软元件算起的 n 点中。



(2) ⓢ 中指定了位数指定的情况下，传送的数据仅为位数指定中指定的位数。

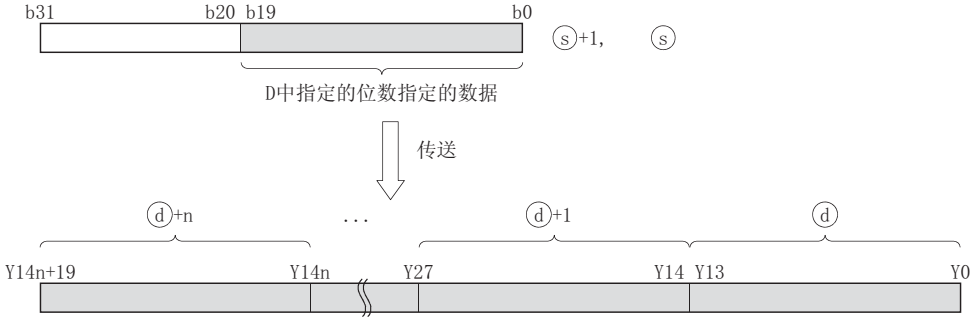
ⓢ 中指定了 K5Y0 的情况下，ⓢ 的字软元件的低 20 位（5 位数）将成为对象。



(3) ⓓ 中指定了位数指定的情况下，ⓓ 中指定的位数指定的位数的数据将被传送。

ⓓ 中指定了 K5Y0 的情况下，ⓢ 的软元件的低 20 位将成为对象。

在ⓢ，ⓓ 二者中指定了位数指定的情况下，与位数无关，ⓓ 中指定的位数指定的数据将被传送。



(4) n 中指定的值为 0 的情况下将变为无处理。

(5) 通过软元件范围检查禁止标志 (SM237)，可以对执行 DFMOV 指令时的软元件范围检查的执行有无进行选择。（仅子设置条件成立时）

出 错

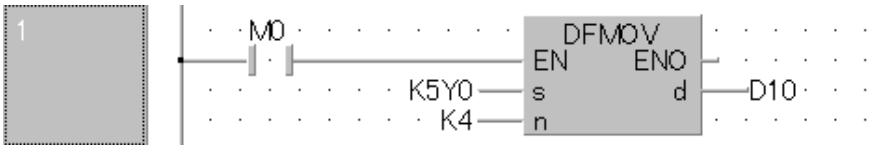
(1) 在以下情况下将变为运算出错状态，出错标志 (SM0) 将变为 ON，出错代码将被存储到 SD0 中。

- n 中指定的数据为负数时。（出错代码： 4100）
- 要传送的数据点数 n 超出了ⓓ的软元件范围时。（出错代码： 4101）

程序示例

(1) 以下为 M0 变为 ON 时，将 Y0 ～ Y13(20 位) 的数据存储到 D10 ～ D17 中的程序。

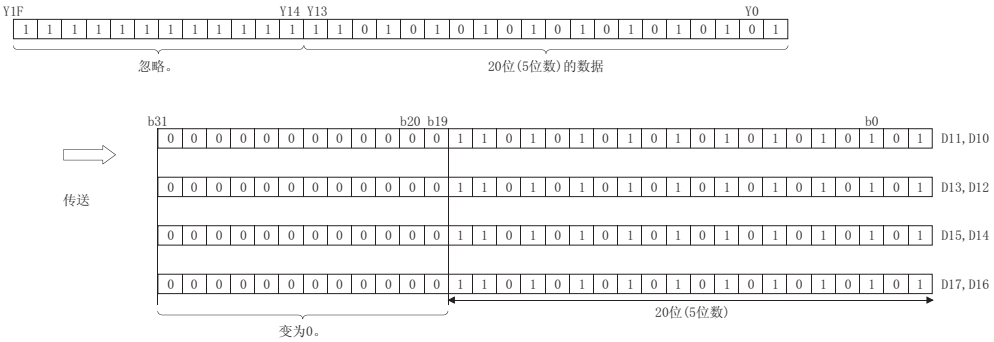
[结构体梯形图]



[ST]

```
DFMOV(M0,K5Y0,K4,D10);
```

[动作]



6.4.9 16 位 /32 位数据替换

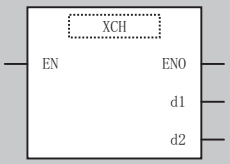
XCH, DXCH

Basic | High performance | Universal | L CPU

XCH(P)
DXCH(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= XCH (EN, d1, d2);

中放入下述指令。

XCH	XCHP
DXCH	DXCHP

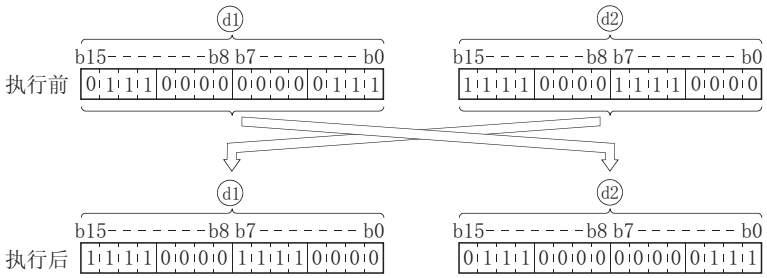
输入自变量，EN: 执行条件 : 位
输出自变量，ENO: 执行结果 : 位
d1: 替换数据 : ANY16/32
d2: 替换数据 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ1				○				-	
ⓓ2				○				-	

功 能

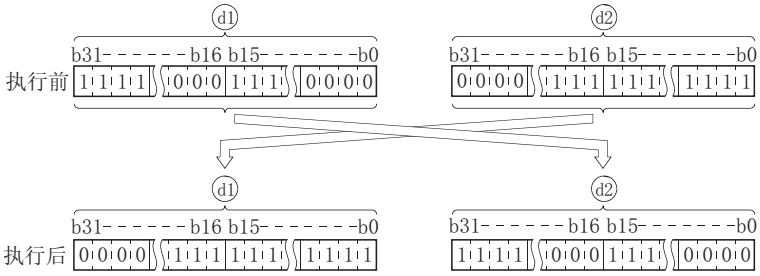
XCH(P)

ⓓ1与ⓓ2的 16 位数据进行替换。



DXCH(P)

将ⓓ1, ⓓ1的 32 位数据进行替换。



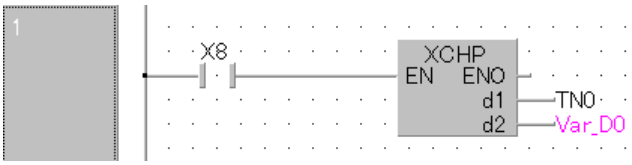
出 错

不存在 XCH(P)、DXCH(P) 指令相关的出错。

程序示例

(1) 以下为 X8 变为 ON 时，将 T0 的当前值与 Var_D0 的内容进行替换的程序。

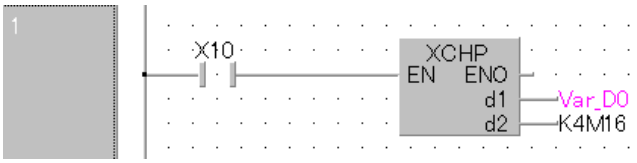
[结构体梯形图]




```
[ST]
XCHP(X8,T0,Var_D0);
```

(2) 以下为 X10 变为 ON 时，将 Var_D0 的内容与 M16 ～ M31 的数据进行替换的程序。

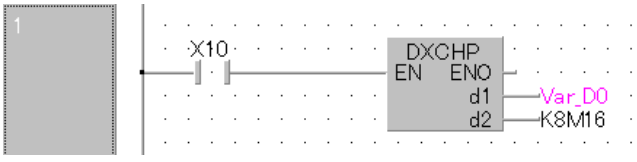
[结构体梯形图]




```
[ST]
XCHP(X10,Var_D0,K4M16);
```

(3) 以下为 X10 变为 ON 时，将 Var_D0 的内容与 M16 ～ M47 的数据进行替换的程序。

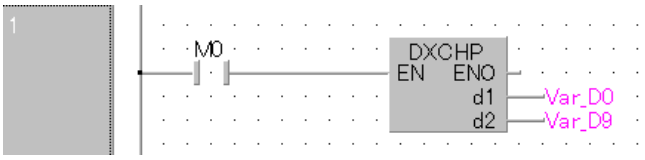
[结构体梯形图]




```
[ST]
DXCHP(X10,Var_D0,K8M16);
```

(4) 以下为 M0 变为 ON 时，将 Var_D0 的内容与 Var_D9 的数据进行替换的程序

[结构体梯形图]



[ST]

```
DXCHP(M0,Var_D0,Var_D9);
```

6.4.10 块16位数据替换

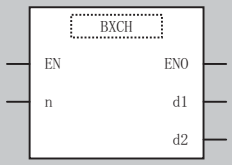
BXCH

Basic | High performance | Universal | L CPU

BXCH(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BXCH (EN, n, d1, d2);

□ 中放入下述指令。

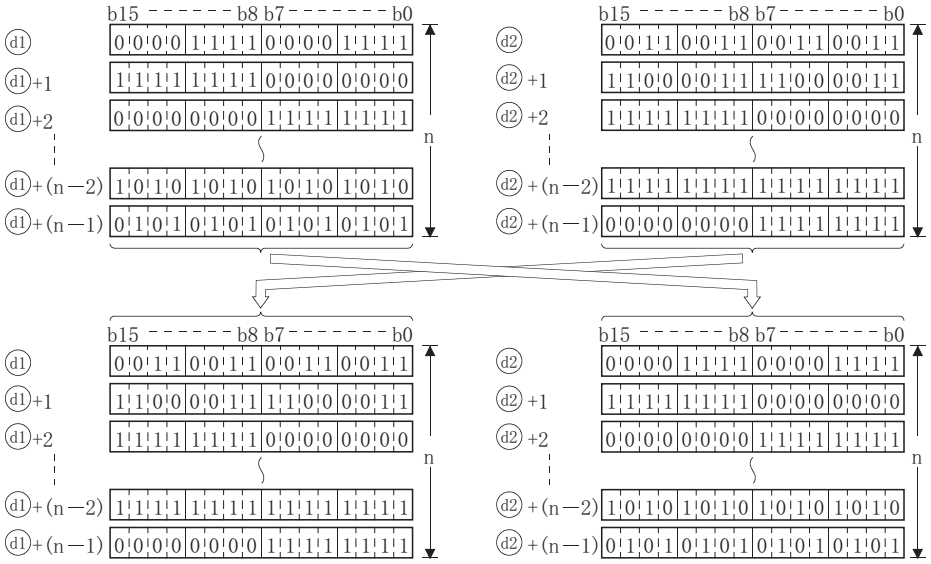
BXCH　　BXCHP

输入自变量，EN:　执行条件　：位
　　n:　替换数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d1:　存储替换数据的软元件的起始编号　：ANY16
　　d2:　存储替换数据的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n	○	○		○					-
ⓓ1	-	○		-					-
ⓓ2	-	○		-					-

功 能

将ⓓ1中指定的软元件算起 n 点的 16 位数据与ⓓ2 中指定的软元件算起 n 点的 16 位数据进行替换。



出 错

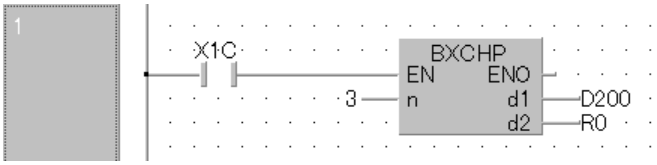
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ1，ⓓ2 的软元件算起 n 点的范围超出了相应软元件时。 (出错代码： 4101)
- ⓓ1，ⓓ2 的软元件重叠时。 (出错代码： 4101)

程序示例

以下为 X1C 变为 ON 时，将 D200 算起 3 点的数据与 R0 算起 3 点的数据进行替换的程序。

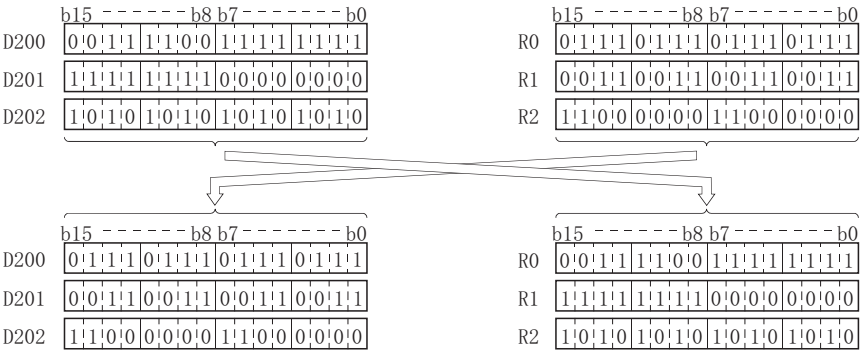
[结构体梯形图]



[ST]

```
BXCHP(X1C,3,D200,R0);
```

[动作]



6.4.11 高低字节替换

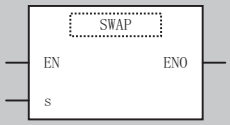
SWAP

Basic | High performance | Universal | L CPU

SWAP(P)

P：执行条件 ：↑

结构体梯形图



ST

ENO:= SWAP (EN, s);

[ ] 中放入下述指令。

SWAP SWAPP

输入自变量， EN: 执行条件 ：位

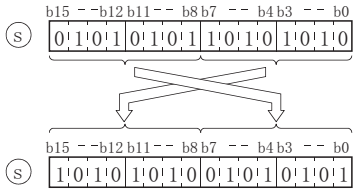
s: 存储替换数据的软元件的起始编号 ：ANY16

输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ			○					-	

功 能

将ⓢ中指定的软元件的高低各 8 位的值进行替换。



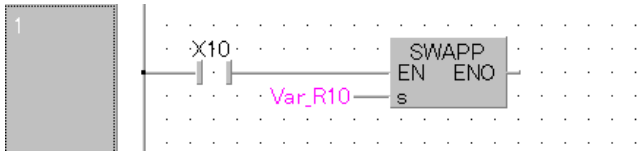
出 错

不存在 SWAP(P) 指令相关的运算出错。

程序示例

以下为 X10 变为 ON 时，将 Var_R10 的高 8 位与低 8 位进行替换的程序。

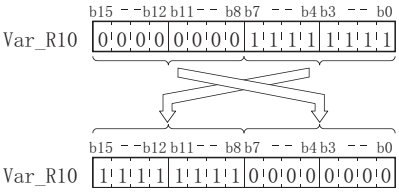
[结构体梯形图]



[ST]

```
SWAPP(X10,Var_R10);
```

[动作]



6.5 程序分支指令

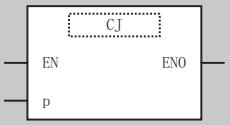
6.5.1 指针分支

CJ, SCJ, JMP

Basic | High performance | Universal | L CPU

CJ
SCJ
JMP

结构体梯形图



ST

不支持。

□ 中放入下述指令。

CJ
SCJ
JMP

输入自变量， EN: 执行条件 : 位
p: 跳转目标的梯形图块标签 : ANY16
输出自变量， ENO: 执行结果 : 位

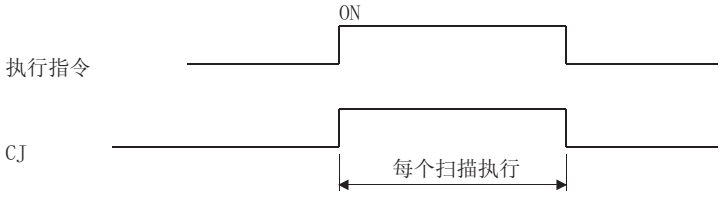
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				P
p				-					○

功 能

CJ

(1) 执行指令为 ON 时，执行同一程序部件内的指定梯形图块标签的程序。

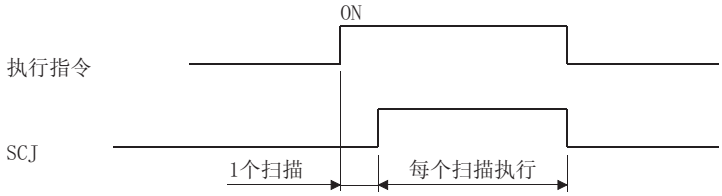
(2) 执行指令为 OFF 时，执行下一步的程序。



SCJ

(1) 从执行指令发生了 OFF → ON 的变化的下一个扫描开始，执行同一程序部件内的指定梯形图块标签的程序。

(2) 执行指令为 OFF 或发生了 ON → OFF 的变化时，执行下一步的程序。



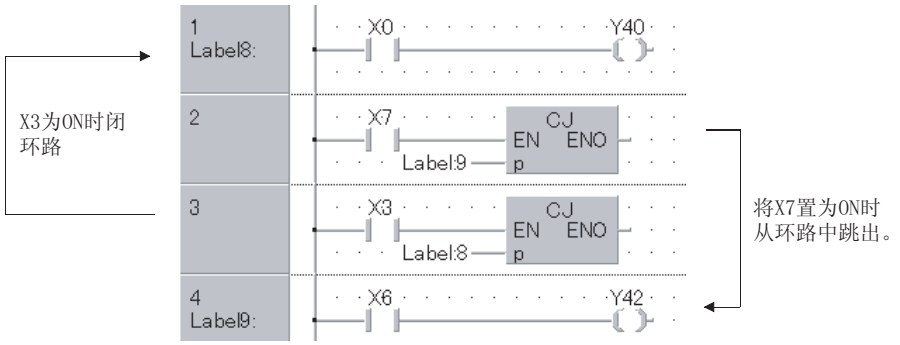
JMP

无条件地执行同一程序部件内的指定梯形图块标签的程序。

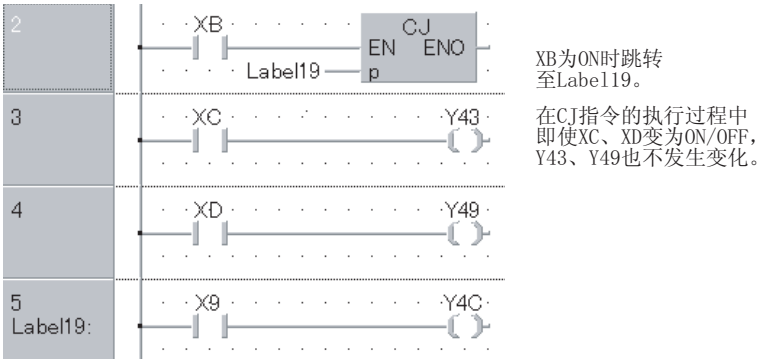
要 点

使用跳转指令时应注意以下几点。

1. 将定时器的线圈置为 ON 后，通过 CJ、SCJ、JMP 指令对线圈为 ON 的定时器进行了跳转的情况下，将无法进行正常的计测。
2. 如果通过 CJ、SCJ、JMP 指令对 OUT 指令进行跳转，扫描时间将变短。
3. 如果通过 CJ、SCJ、JMP 指令跳转到后面，扫描时间将变短。
4. 通过 CJ、SCJ、JMP 指令可以从执行中的步跳转到小编号的步。但是，为了避免看门狗定时器的时间到，需要考虑从此期间环路中跳出的方法。



5. 通过 CJ、SCJ、JMP 指令跳过的软元件不发生变化。



6. 跳转指令只能指定同一程序部件内的梯形图块标签。

出 错

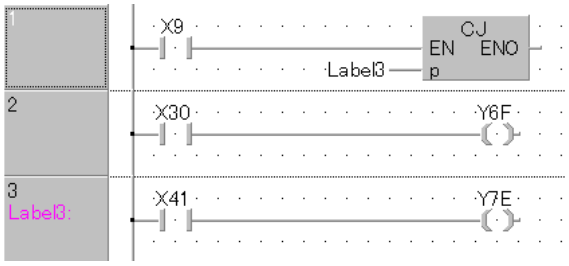
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定的指针编号在 END 指令以前不存在时。（出错代码：4210）
- 在同一程序中指定了未作为标签使用的指针编号时。（出错代码：4210）
- 指定了其它程序中的某个公共指针时。（出错代码：4210）

程序示例

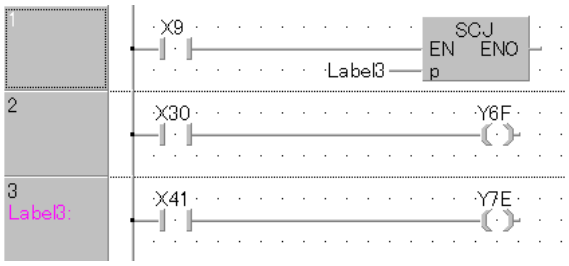
(1) 以下为 X9 变为 ON 时跳转至 Label3 的程序。

[结构体梯形图]



(2) 以下为从 X9 变为 ON 的下一个扫描开始跳转至 Label3 的程序。

[结构体梯形图]

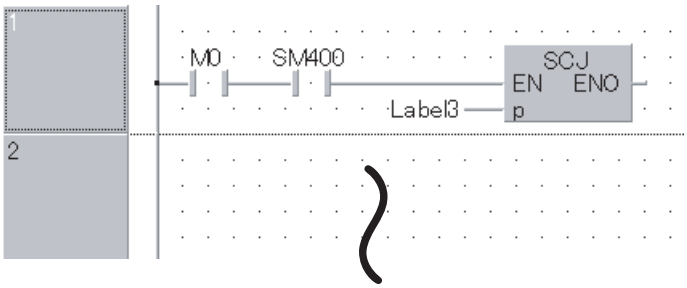


注意事项

(1) 在通用型 QCPU、LCPU 中使用 SCJ 指令时，需要在 SCJ 指令之前插入 AND SM400。

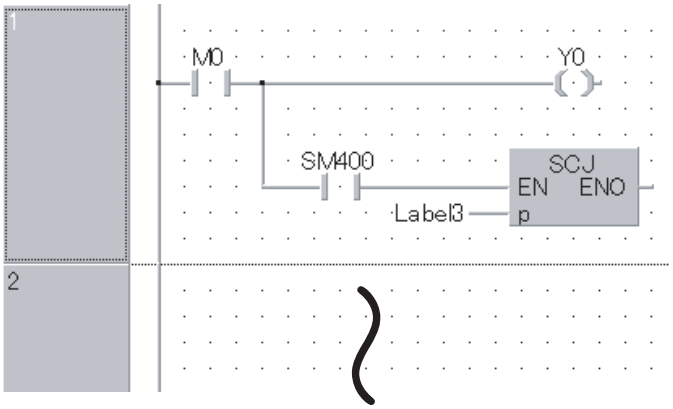
[程序示例 1]

[结构体梯形图]



[程序示例 2]

[结构体梯形图]



6.5.2 跳转至 END

GOEND

Basic | High performance | Universal | L CPU

GOEND

结构体梯形图

ST

```
ENO:= GOEND (EN);
```

[ ] 中放入下述指令。

GOEND

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

跳转至同一程序文件内的 FEND 指令处。

出 错

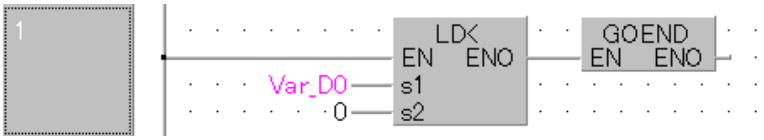
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 CALL 指令后，在执行 RET 指令之前执行了 GOEND 指令时。 (出错代码： 4211)
- 执行 FOR 指令后，在执行 NEXT 指令之前执行了 GOEND 指令时。 (出错代码： 4200)
- 在中断程序中在执行 IRET 指令之前执行了 GOEND 指令时。 (出错代码： 4221)
- 在 CHKCIR ～ CHKEND 指令内执行了 GOEND 指令时。 (出错代码： 4230)
- 在 IX ～ IXEND 指令内执行了 GOEND 指令时。 (出错代码： 4231)

程序示例

以下为 Var_D0 的内容为负数时，跳转至 END 的程序。

[结构体梯形图]



[ST]

```
GOEND(Var_D0<0);
```

6.6 程序执行控制指令

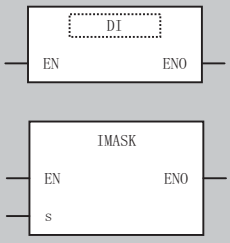
6.6.1 中断禁止 / 允许，中断程序屏蔽

DI, EI, IMASK

Basic | High performance | Universal | L CPU

DI
IMASK
EI

结构体梯形图



ST

ENO:= DI (EN);

ENO:=IMASK(EN, s);

中放入下述指令。

DI
EI

输入自变量， EN: 执行条件 : 位
s: 中断屏蔽数据 (IMASK 时) :ANY16 的数组 (0..15)
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	-	○		-					

1 使用基本型 QCPU 时

功 能

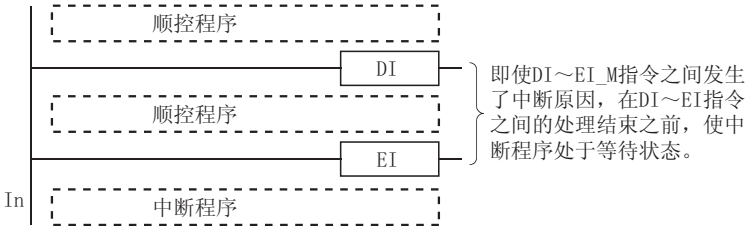
DI

(1) 即使发生了中断程序的启动原因，在执行 EI 指令之前禁止中断程序的执行。

(2) 投入电源时或对 CPU 模块进行了复位的情况下，变为 DI 状态。

EI

对执行 DI 指令时的中断禁止状态进行解除，将通过 IMASK 指令置为允许的中断指针编号的中断程序及恒定周期执行类型程序置为允许执行状态。
IMASK 指令非执行时，I32 ～ I47 将变为中断禁止状态。



IMASK

(1) 根据ⓢ中指定的软元件算起 8 点的位模式，将指定的中断指针编号的中断程序置为执行允许状态 / 执行禁止状态。

- 1(ON) ..... 中断程序的执行允许状态
- 0(OFF) .... 中断程序的执行禁止状态

(2) 对应于各个位的中断指针编号如下所示。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ⓢ	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
ⓢ+1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
ⓢ+2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
ⓢ+3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
ⓢ+4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
ⓢ+5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
ⓢ+6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
ⓢ+7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112

(3) 投入电源时或对 CPU 模块进行了复位的情况下，I0 ～ I31、I48 ～ I127 的中断程序将变为执行允许状态。此外，I32 ～ I47 的中断程序将变为执行禁止状态。

(4) ⓢ, ⓢ+1, ⓢ+2, ⓢ+3 ～ⓢ+7 的软元件状态被存储到 SD715 ～ SD717、SD781 ～ SD785(IMASK 指令屏蔽模式存储区域) 中。

(5) 对于特殊寄存器，被分割为 SD715 ～ SD717, SD781 ～ SD785 的状态，但对于ⓢ～ⓢ+7 的软元件编号，应进行连续设置。

要 点

1. 关于中断条件，请参阅所使用的 CPU 模块的用户手册（功能解解说 / 程序基础篇）。
2. 中断程序中处于 DI（中断禁止）状态。不要在中断程序中插入 EI 指令，不要进行多重中断，即在中断程序的执行过程中不要执行其它中断程序。
3. 主控过程中存在有 EI、DI 指令时，与 MC 指令的执行、非执行无关，执行 EI、DI 指令。

出 错

不存在 DI、EI、IMASK 指令相关的运算出错。

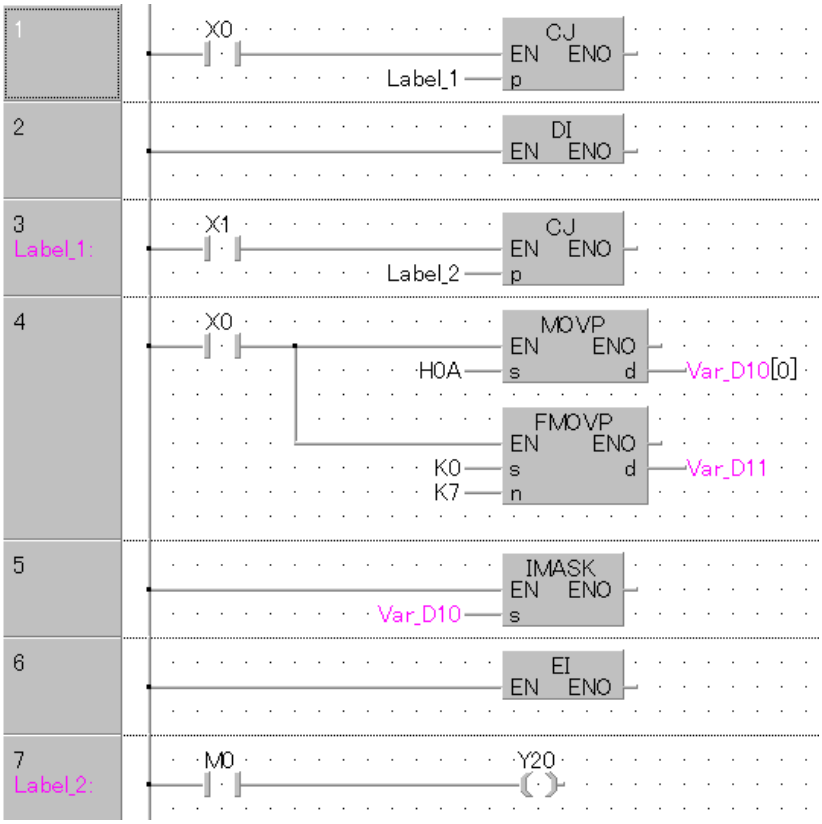
程序示例

以下为 X0 处于 ON 状态时，将中断指针编号 I1、I3 的中断程序置为执行允许状态的程序。

[结构体梯形图]

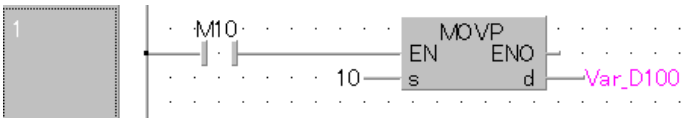
Task_01[常时执行]…中断允许程序

无需 FEND 指令。



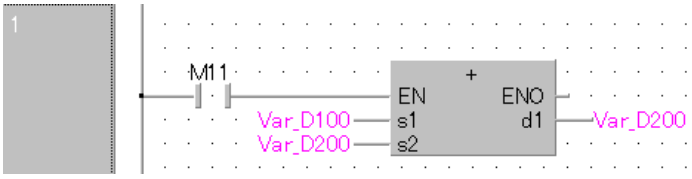
Task_ 中断 No1[事件启动]…I1 的中断允许程序

无需 IRET 指令。



Task_ 中断 No3[事件启动]…I3 的中断允许程序

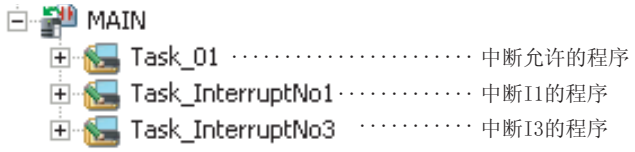
无需 IRET 指令。



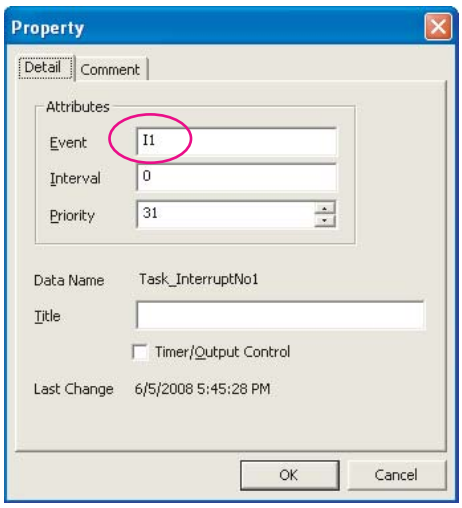
```
[ST]
Task_01[ 常时执行 ]…中断允许程序
IF (X0=FALSE) THEN
     DI(TRUE);
END_IF;
IF (X0=FALSE) THEN
     IF X0 THEN
          MOVP(TRUE,H0A,D10);
          FMOVP(TRUE,0,7,D11);
     END_IF;
     IMASK(TRUE,D10);
     EI(TRUE);
END_IF;
OUT(M0,Y20);

Task_ 中断 No1[ 事件启动 ]…I1 的中断允许程序
MOVP(M10,10,Var_D100);

Task_ 中断 No3[ 事件启动 ]…I3 的中断允许程序
IF M11 THEN
     Var_D200 := Var_D100 + Var_D200;
END_IF;
```

要 点

创建中断程序时，创建中断允许程序及用于登录中断程序的多个任务。



将 Task_ 中断 No1[常时执行] 变更为 [事件启动]…中断程序。
将 Task_ 中断 No3[常时执行] 变更为 [事件启动]…中断程序。
[属性设置]



输入软元件I1/I3。

2 使用高性能型 QCPU/ 通用型 QCPU/LCPU 时

功 能

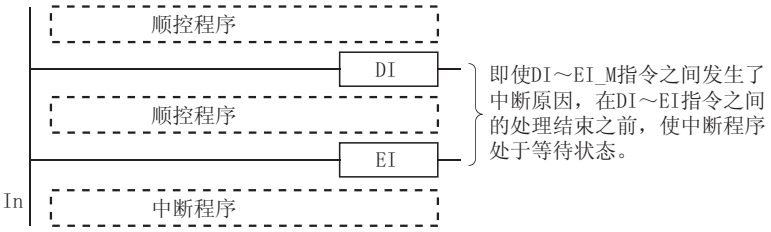
DI

(1) 即使发生了中断程序的启动原因，在执行 EI 指令之前禁止中断程序的执行。

(2) 投入电源时或对 CPU 模块进行了复位的情况下，变为 DI 状态。

EI

对执行 DI 指令时的中断禁止状态进行解除，将通过 IMASK 指令置为允许的中断指针编号的中断程序及恒定周期执行类型程序置为允许执行状态。
IMASK 指令非执行时，I32 ～ I47 将变为中断禁止状态。



IMASK

(1) 根据Ⓢ中指定的软元件算起 16 点的位模式，将指定的中断指针编号的中断程序置为执行允许状态 / 执行禁止状态。

- 1(ON) ..... 中断程序的执行允许状态
- 0(OFF) .... 中断程序的执行禁止状态

(2) 对应于各个位的中断指针编号如下所示。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ⓢ	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
ⓢ+1	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
ⓢ+2	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
ⓢ+3	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
ⓢ+4	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
ⓢ+5	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
ⓢ+6	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
ⓢ+7	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112
ⓢ+8	I143	I142	I141	I140	I139	I138	I137	I136	I135	I134	I133	I132	I131	I130	I129	I128
ⓢ+9	I159	I158	I157	I156	I155	I154	I153	I152	I151	I150	I149	I148	I147	I146	I145	I144
ⓢ+10	I175	I174	I173	I172	I171	I170	I169	I168	I167	I166	I165	I164	I163	I162	I161	I160
ⓢ+11	I191	I190	I189	I188	I187	I186	I185	I184	I183	I182	I181	I180	I179	I178	I177	I176
ⓢ+12	I207	I206	I205	I204	I203	I202	I201	I200	I199	I198	I197	I196	I195	I194	I193	I192
ⓢ+13	I223	I222	I221	I220	I219	I218	I217	I216	I215	I214	I213	I212	I211	I210	I209	I208
ⓢ+14	I239	I238	I237	I236	I235	I234	I233	I232	I231	I230	I229	I228	I227	I226	I225	I224
ⓢ+15	I255	I254	I253	I252	I251	I250	I249	I248	I247	I246	I245	I244	I243	I242	I241	I240

(3) 投入电源时或对 CPU 模块进行了复位的情况下，I0 ～ I31、I48 ～ I255 的中断程序将变为执行允许状态。此外，I32 ～ I47 的中断程序将变为执行禁止状态。

(4) ⓢ，ⓢ+1，ⓢ+2，ⓢ+3 ～ⓢ+15 的软元件状态被存储到 SD715 ～ SD717，SD781 ～ SD793(IMASK 指令屏蔽模式存储区域）中。

(5) 对于特殊寄存器，被分割为 SD715 ～ SD717、SD781 ～ SD793 的状态，但对于ⓢ～ⓢ+15 的软元件编号，应进行连续设置。

要 点

1. 关于中断条件，请参阅所使用的 CPU 模块的用户手册（功能解解说 / 程序基础篇）。
2. 中断程序中处于 DI（中断禁止）状态。不要在中断程序中插入 EI 指令，不要进行多重中断，即在中断程序的执行过程中不要执行其它中断程序。
3. 主控过程中存在有 EI、DI 指令时，与 MC 指令的执行、非执行无关，执行 EI、DI 指令。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的软元件超出了相应软元件的范围时。（通用型 QCPU、LCPU 时）
  （出错代码： 4101）

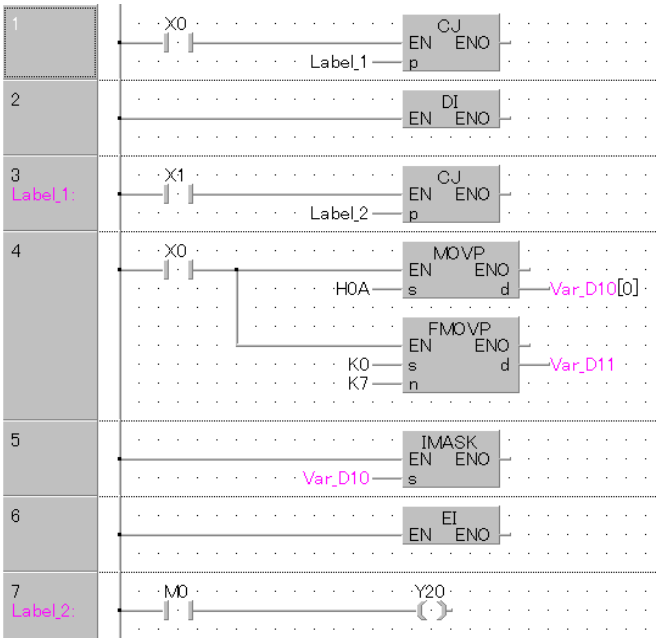
程序示例

以下为 X0 处于 ON 状态时，将中断指针编号 I1、I3 的中断程序置为执行允许状态的程序。

[结构体梯形图]

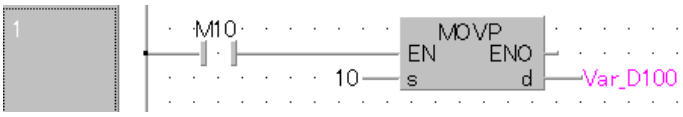
Task_01[常时执行]…中断允许程序

无需 FEND 指令。



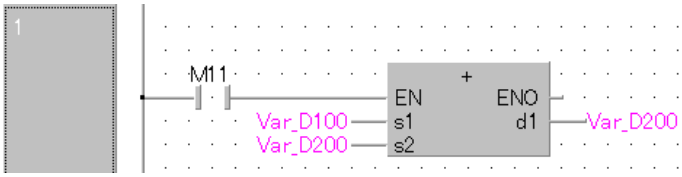
Task_ 中断 No1[事件启动]…I1 的中断允许程序

无需 IRET 指令。



Task_ 中断 No3[事件启动]…I3 的中断允许程序

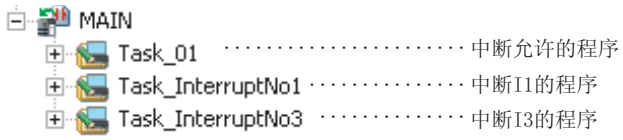
无需 IRET 指令。



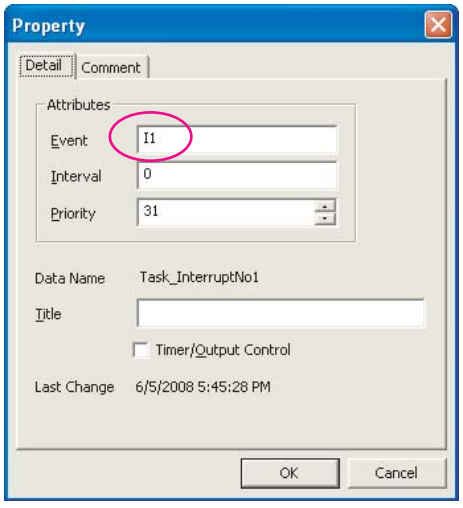
```
[ST]
Task_01[ 常时执行 ]…中断允许程序
IF (X0=FALSE) THEN
     DI(TRUE);
END_IF;
IF (X0=FALSE) THEN
     IF X0 THEN
          MOVP(TRUE,H0A,D10);
          FMOVP(TRUE,0,15,D11);
     END_IF;
     IMASK(TRUE,D10);
     EI(TRUE);
END_IF;
OUT(M0,Y20);

Task_ 中断 No1[ 事件启动 ]…I1 的中断允许程序
MOVP(M10,10,Var_D100);

Task_ 中断 No3[ 事件启动 ]…I3 的中断允许程序
IF M11 THEN
     Var_D200 := Var_D100 + Var_D200;
END_IF;
```

要 点

创建中断程序时，创建中断允许程序及用于登录中断程序的多个任务。



将 Task_ 中断 No1[常时执行] 变更为 [事件启动]…中断程序。
将 Task_ 中断 No3[常时执行] 变更为 [事件启动]…中断程序。
[属性设置]



输入软元件I1/I3。

6.6.2 从中断程序返回

IRET

Basic | High performance | Universal | L CPU

IRET

结构体梯形图

ST

ENO:= IRET (EN);

中放入下述指令。

IRET

输入自变量， EN: 执行条件 :位

输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

(1) 显示中断程序处理的结束。

(2) 编译时将被自动附加，因此无需使用。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 不存在对应于中断 No. 的指针时。 (出错代码： 4220)
- 在执行中断程序之前，执行了 IRET 指令时。 (出错代码： 4223)
- 发生中断后执行 IRET 指令之前，执行了 END、FEND、GOEND、STOP 指令时。 (出错代码： 4221)
- 在恒定周期执行类型程序中执行了 IRET 指令时。 (通用型 QCPU、LCPU 时) (出错代码： 4223)

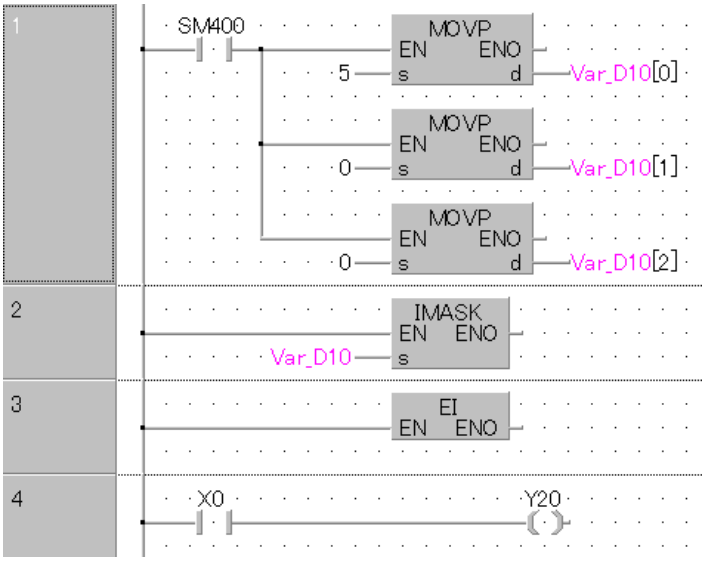
程序示例

以下为发生了了 No.3 的中断时，如果 M0 为 ON 则对 Var_D0 进行 +1 的程序。

[结构体梯形图]

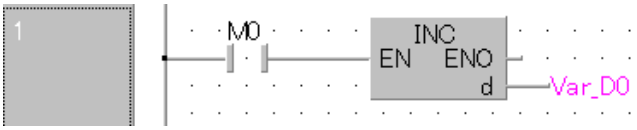
Task_01[常时执行]…中断允许程序

中断程序的情况下，无需 FEND 指令。



Task_ 中断 No.3[事件启动]…中断允许程序

中断程序的情况下，无需 IRET 指令。



[ST]

Task_01[常时执行]…中断允许程序

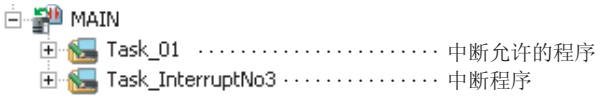
```
IF SM400 THEN
     MOVP(TRUE,H5,Var_D10[0]),
     MOVP(TRUE,H0,Var_D10[1]);
     MOVP(TRUE,H0,Var_D10[2]);
END_IF;
IMASK(SM400,Var_D10);
EI(SM400);
OUT(X0,Y20);
```

Task_ 中断 No.3[事件启动]…中断程序

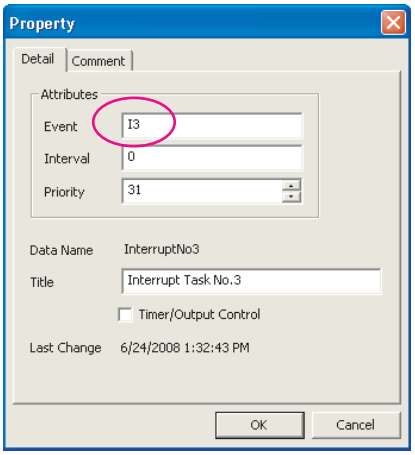
```
INC(M0,Var_D0);
```

要 点

创建中断程序时，创建中断允许程序及用于登录中断程序的多个任务。



将 Task_ 中断 No3[常时执行] 变更为 [事件启动]…中断程序。
[属性设置]



输入软元件I3。

6.7 I/O刷新指令

6.7.1 I/O刷新指令

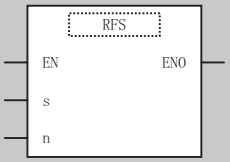
RFS

Basic | High performance | Universal | L CPU

RFS(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= RFS (EN, s, n);

□ 中放入下述指令。

RFS　　RFSP

输入自变量，EN:　执行条件　：位
　　s:　进行刷新的软元件的起始编号　：位
　　n:　刷新点数　：ANY16
输出自变量，ENO:　执行结果　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
Ⓢ	○(仅X,Y)		-						-
n	○		○						-

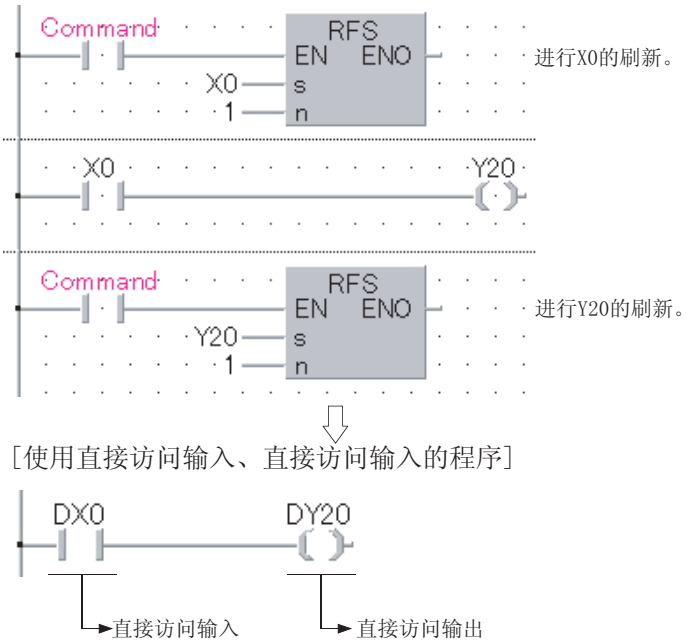
功 能

(1) 是在1个扫描过程中仅对相应的软元件进行刷新，进行外部输入的获取或至输出模块的输出的功能。

(2) 输入的获取及至外部的输出是仅在程序的END执行后批量地进行，因此1个扫描中不能向外部输出脉冲信号。
如果执行I/O刷新指令，在程序执行过程中对相应的输入(X)或输出(Y)进行强制刷新，因此1个扫描中可以向外部输出脉冲信号。

(3) 对输入 (X) 或输出 (Y) 以 1 点为单位进行刷新的情况下，应使用直接访问输入 (DX)、直接访问输出 (DY)。

[使用RFS指令的程序]



[使用直接访问输入、直接访问输入的程序]

出 错

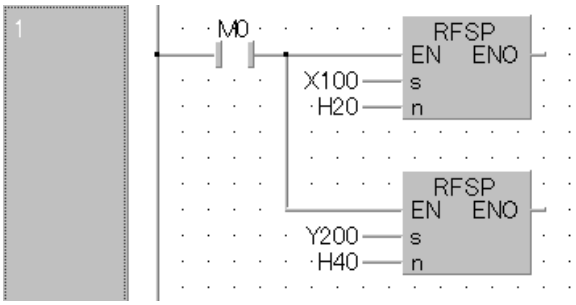
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件算起 n 点的范围超出了相邻 I/O 的范围时。　　(出错代码： 4101)

程序示例

以下为 M0 变为 ON 时，对 X100 ～ X11F、Y200 ～ Y23F 进行刷新的程序。

[结构体梯形图]




```
[ST]
IF M0 THEN
     RFSP(TRUE,X100,H20);
     RFSP(TRUE,Y200,H40);
END_IF;
```

6.8 其它方便指令

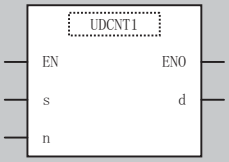
6.8.1 单相输入增 / 减计数器

UDCNT1

Basic | High performance | Universal | L CPU

UDCNT1

结构体梯形图



ST

ENO:= UDCNT1 (EN, s, n, d);

[ ] 中放入下述指令。

UDCNT1

输入自变量， EN: 执行条件 : 位

s: s+0: 计数输入用的输入编号 : 位的数组 (0..1)

s+1: 增 / 减设置用

• OFF.... 递增计数

• ON..... 递减计数

n: 设置数据拘 : ANY16

输出自变量， ENO: 执行结果 : 位

d: 通过 UDCNT1 指令进行计数的计数器编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○(仅X)*1	–	–	–					–
n	△*2	△*2	△*2	○					–
ⓓ	–	△*1(仅C)	–	–					–

*1: 在全局标签中，应指定在软元件中设置了 X 的数组。X 软元件只能在输入输出点数（允许访问实际输入输出模块的点数）的范围内使用。

*2: 不能使用局部软元件及各程序中设置的文件寄存器。

功 能

(1) ⓢ中指定的输入由 OFF → ON 时，对ⓓ中指定的计数器的当前值进行更新。

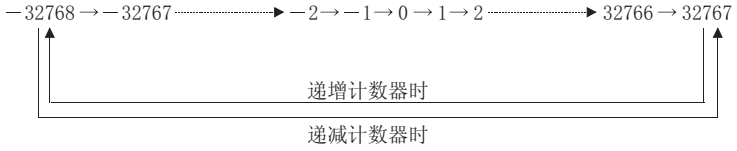
(2) 计数方向取决于ⓢ+1 中指定的输入的 ON/OFF。

• OFF : 递增计数（将当前值向增加方向计数）

• ON : 递减计数（将当前值向减少方向计数）

(3) 计数处理的执行情况如下所示。

- 在递增计数中，当前值等于 n 中指定的设置值时，ⓓ 中指定的计数器的触点将变为 ON。
  但是，即使ⓓ 中指定的计数器的触点变为 ON，当前值的计数仍将继续。（参阅程序示例 (1)）
- 在递减计数中，当前值变为设置值 -1 时，ⓓ 中指定的计数器的触点将变为 OFF。（参阅程序示例 (1)）
- ⓓ 中指定的计数器为环形计数器。
  当前值为 32767 时如果进行递增计数，则当前值将变为 -32768。
  此外，当前值为 -32768 时如果进行递减计数，则当前值将变为 32767。
  当前值的计数处理内容如下所示。



(4) 对于通过 UDCNT1 进行的计数处理，在执行指令 OFF → ON 时开始计数，ON → OFF 时中止计数。
再次将执行指令置为 OFF → ON 时，将从中止时的当前值开始进行计数。

(5) ⓓ 中指定的计数器的当前值清除及触点的 OFF 是通过 RST 指令进行。

要 点

1. 执行 UDCNT1 指令时，将自变量的软元件数据登录到 CPU 模块的工作区，实际的计数动作是通过系统中断进行处理。
   （对于 CPU 模块的工作区中登录的软元件数据，在将执行指令置为 OFF 或 STOP → RUN 时将被清除。）
   因此对于可计数的脉冲，其 ON 及 OFF 时间必须大于 CPU 模块的中断间隔时间。
   CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、通用型 QCPU、LCPU	1ms

2. 通过 UDCNT1 指令进行计数的过程中（执行指令处于 ON 状态），不能对设置值进行变更。
   对设置值进行变更时，应在将执行指令置为 OFF 之后进行。
3. 对于在 UDCNT1 指令中指定的计数器，不能在其它指令中使用。
   如果在其它指令中使用，将无法正常进行计数。
4. UDCNT1 指令在处于执行状态的全部程序中最多可以使用 6 次。
   第 7 次及以后的 UDCNT1 指令将变为无处理。

出 错

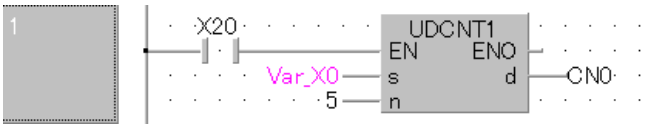
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件超出了相应软元件的范围时。　　（出错代码： 4101）

程序示例

以下为将 X20 变为 ON 之后的 Var_X0 的 OFF → ON 的次数，通过 C0（增 / 减计数器）进行计数的程序。

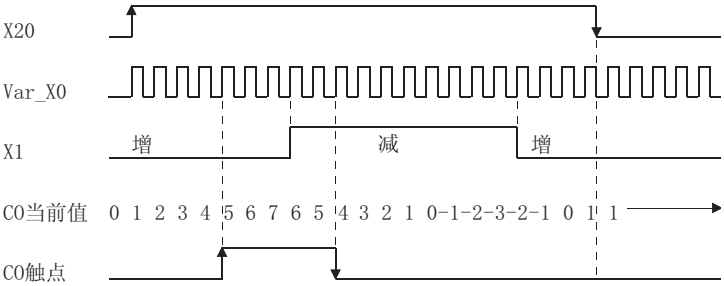
[结构体梯形图]



[ST]

```
UDCNT1(X20,Var_X0,5,C0);
```

[动作]



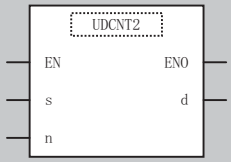
6.8.2 2 相输入增 / 减计数器

UDCNT2

Basic | High performance | Universal | L CPU

UDCNT2

结构体梯形图



ST

```
ENO:=  UDCNT2  (EN, s,  n,  d);
```

[ ] 中放入下述指令。

UDCNT2

输入自变量， EN: 执行条件 : 位

s: s+0: 计数输入用的输入编号 (A 相脉冲) : 位的数组 (0..1)

s+1: 计数输入用的输入编号 (B 相脉冲)

n: 设置数据拘 : ANY16

输出自变量， ENO: 执行结果 : 位

d: 通过 UDCNT2 指令进行计数的计数器编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○ (仅X仅)*1	–	–	–					–
n	△ *2	△ *2	△ *2	○					–
ⓓ	–	△*2(仅C)	–	–					–

*1: 在全局标签中，应指定在软元件中设置了 X 的数组。X 软元件只能在输入输出点数 (允许访问实际输入输出模块的点数) 的范围内使用。

*2: 不能使用局部软元件及各程序中设置的文件寄存器。

功 能

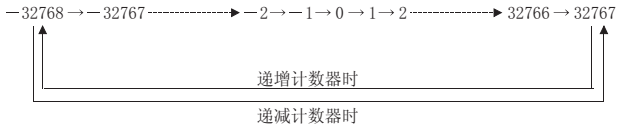
(1) 根据ⓢ中指定的输入 (A 相脉冲) 及ⓢ+1 中指定的输入 (B 相脉冲) 的状态，对ⓓ中指定的计数器的当前值进行更新。

(2) 计数方向按以下方式确定。

- ⓢ为 ON 时ⓢ+1 由 OFF → ON 的情况下，进行递增计数 (将当前值向增加方向计数)。
- ⓢ为 ON 时ⓢ+1 由 ON → OFF 的情况下，进行递减计数 (将当前值向减少方向计数)。
- ⓢ为 OFF 时不进行计数。

(3) 计数处理的执行情况如下所示。

- 在递增计数中，当前值等于 n 中指定的设置值时，ⓓ 中指定的计数器的触点将变为 ON。
  但是，即使ⓓ 中指定的计数器的触点变为 ON，当前值的计数仍将继续。
  (参阅程序示例 (1))
- 在递减计数中，当前值变为设置值 -1 时，ⓓ 中指定的计数器的触点将变为 OFF。(参阅程序示例 (1))
- ⓓ 中指定的计数器为环形计数器。
  当前值为 32767 时如果进行递增计数，则当前值将变为 -32768。
  此外，当前值为 -32768 时如果进行递减计数，则当前值将变为 32767。
  当前值的计数处理内容如下所示。



(4) 对于通过 UDCNT2 进行的计数处理，在执行指令 OFF → ON 时开始计数，ON → OFF 时中止计数。
再次将执行指令置为 OFF → ON 时，将从中止时的当前值开始进行计数。

(5) ⓓ 中指定的计数器的当前值清除及触点的 OFF 是通过 RST 指令进行。

要 点

1. 执行 UDCNT2 指令时，将自变量的软元件数据登录到 CPU 模块的工作区，实际的计数动作是通过系统中断进行处理。
   (对于 CPU 模块的工作区中登录的软元件数据，在将执行指令置为 OFF 或 STOP → RUN 时将被清除。)
   因此对于可计数的脉冲，其 ON 及 OFF 时间必须大于 CPU 模块的中断间隔时间。
   CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、通用型 QCPU、LCPU	1ms

2. 通过 UDCNT2 指令进行计数的过程中 (执行指令处于 ON 状态)，不能对设置值进行变更。
   对设置值进行变更时，应将执行指令置为 OFF 之后进行。
3. 对于在 UDCNT2 指令中指定的计数器，不能在其它指令中使用。
   如果在其它指令中使用，将无法正常进行计数。
4. UDCNT2 指令在处于执行状态的全部程序中最多可以使用 5 次。
   第 6 次及以后的 UDCNT1 指令将无处理。

出 错

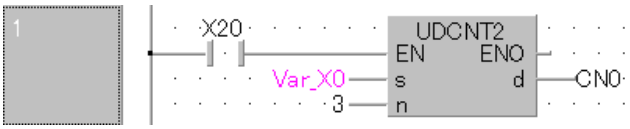
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件超出了相应软元件的范围时。 (出错代码 : 4101)

程序示例

以下为将 X20 变为 ON 之后的 Var_X0、Var_X0+1 的状态通过 C0（增 / 减计数器）进行计数的程序。

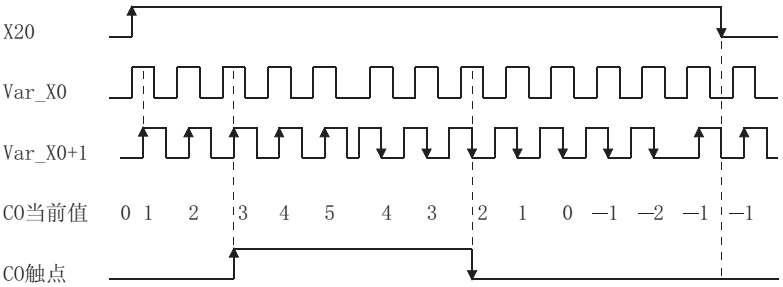
[结构体梯形图]



[ST]

```
UDCNT2(X20,Var_X0,3,CN0);
```

[动作]



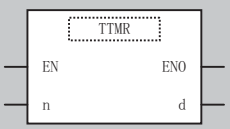
6.8.3 示教定时器

TTMR

Basic | High performance | Universal | L CPU

TTMR

结构体梯形图



ST

```
ENO:= TTMR (EN, n, d);
```

□ 中放入下述指令。

TTMR

输入自变量， EN: 执行条件 : 位

n: 测定值的乘数 : ANY16

输出自变量， ENO: 执行结果 : 位

d: d+0: 测定值存储软元件 : ANY16 的数组 (0..1)

d+1:CPU 模块的系统用

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n	○	○		○					-
ⓓ	-	○		-					-

功 能

(1) 对执行指令的 ON 状态时间以秒为单位进行测定后，将其与 n 中指定的乘数相乘后的值存储到 ⓓ 中指定的软元件中。

(2) 执行指令由 OFF → ON 时，对 ⓓ +0, ⓓ +1 中指定的软元件进行清除。

(3) n 中可指定的乘数如下所示。

n	乘数
0	1
1	10
2	100

要 点

1. 执行 TTMR 指令执行时，进行时间计测。
   将 TTMR 指令通过 JMP 指令等进行了跳过的情况下，将无法进行正确的测定。
2. 在 TTMR 指令的执行过程中，不要对 n 中指定的乘数进行变更。
   如果对 n 中指定的乘数进行了变更，将无法得到正确的值。
3. TTMR 指令在低速执行类型程序中也可使用。
4. ⓓ+1 中指定的软元件是为 CPU 模块的系统所使用，用户不要对该值进行变更。
   如果用户对该值进行了变更，ⓓ中指定的软元件中存储的值将变为不正确的值。

(4) n 中指定的值为 0 ～ 2 以外时将变为无处理。

出 错

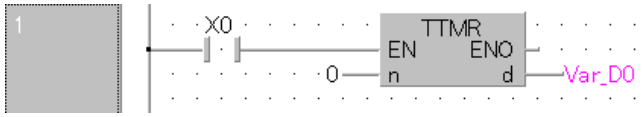
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时)　　(出错代码： 4101)

程序示例

以下为将 X0 处于 ON 状态的时间存储到 Var_D0 中的程序。

[结构体梯形图]



[ST]

```
TTMR(X0,0,Var_D0);
```

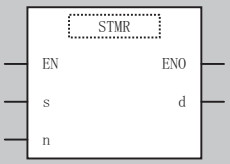
6.8.4 特殊功能定时器

STMR

Basic | High performance | Universal | L CPU

STMR

结构体梯形图



ST

ENO:= STMR (EN, s, n, d);

[ ] 中放入下述指令。

STMR

输入自变量， EN: 执行条件 : 位

s: 定时器编号 : ANY16

n: 设置值 : ANY16

输出自变量， ENO: 执行结果 : 位

d: d[0]:OFF 延迟定时器输出 : 位的数组 (0..3)

d[1]:OFF 后单次定时器输出

d[2]:ON 后单次定时器输出

d[3]:ON 延迟 +OFF 延迟定时器输出

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	–	△*1	–	–					–
n	○	○	○	○					–
ⓓ	○	–	–	–					–

*1 ：只能使用定时器 (T)。

功 能

(1) 在 STMR 指令中，使用ⓓ中指定的软元件算起的 4 点，进行 4 种类型的定时器输出。

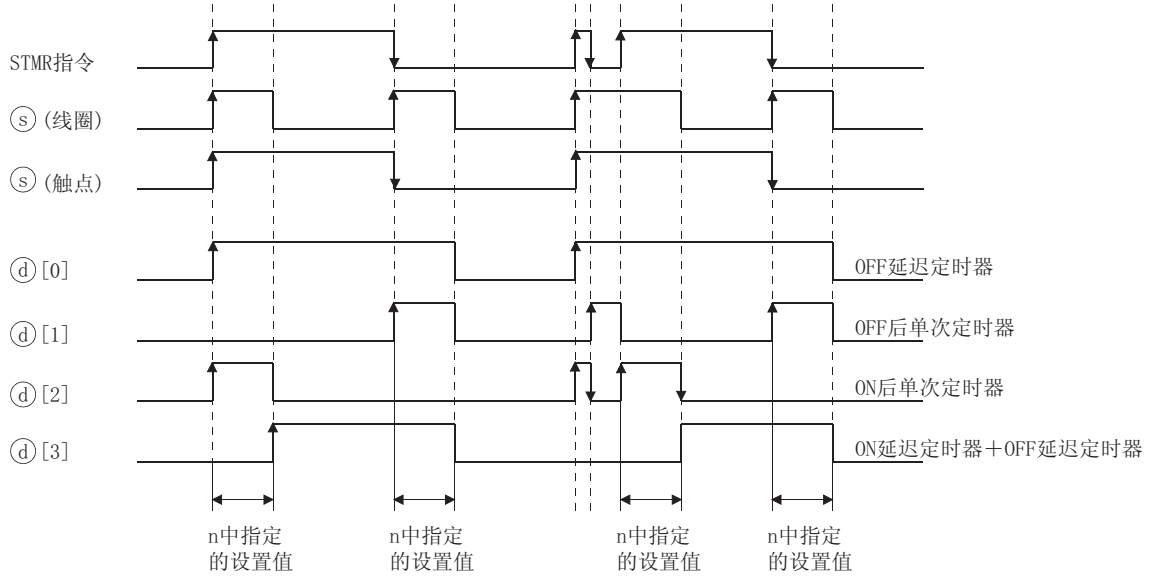
- OFF 延迟定时器输出 (ⓓ+0)
  在 STMR 指令的上升沿时变为 ON，在指令的下降沿且经过了 n 中指定的时间后变为 OFF。
- OFF 后单次定时器输出 (ⓓ+1)
  在 STMR 指令的下降沿时变为 ON，在经过了 n 中指定的时间后变为 OFF。
- ON 后单次定时器输出 (ⓓ+2)
  在 STMR 指令的上升沿时变为 ON，在经过了 n 中指定的时间或 STMR 指令为 OFF 时变为 OFF。

- ON 延迟 +OFF 延迟定时器输出（ⓓ+3）
  在定时器线圈的下降沿时变为 ON，在 STMR 指令的下降沿且经过了 n 中指定的时间后变为 OFF。

(2) 对于ⓢ中指定的定时器的线圈，在 STMR 指令的上升沿或下降沿时变为 ON，开始进行当前值计测。

- 对于定时器的线圈，在到达 n 中指定的设置值之前进行计测，在时间到的情况下变为 OFF。
- 对于定时器的线圈，在时间到之前如果 STMR 指令变为 OFF，则将保持为 ON 状态不变。此时定时器的计测将中止。
  如果再次将 STMR 指令置为 ON，则将当前值置为 0 后，重新开始计测。

(3) 定时器的触点在 STMR 指令的上升沿时变为 ON，在定时器线圈的下降沿之后，在 STMR 指令的下降沿时变为 OFF。
定时器的触点是为 CPU 模块的系统所用，因此用户不能使用。



(4) 执行 STMR 指令时，对 STMR 指令中指定的定时器的当前值进行计测。
通过 JMP 指令等跳过了 STMR 指令的情况下，将无法进行正常计测。

(5) ⓓ中指定的定时器的计测单位与低速定时器相同。

(6) n 的设置值的允许指定范围为 1 ～ 32767。

(7) 对于ⓢ中指定的定时器，不能在 OUT 指令中使用。
在 STMR 指令与 OUT 指令中使用了相同的定时器编号的情况下，将无法正常执行动作。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

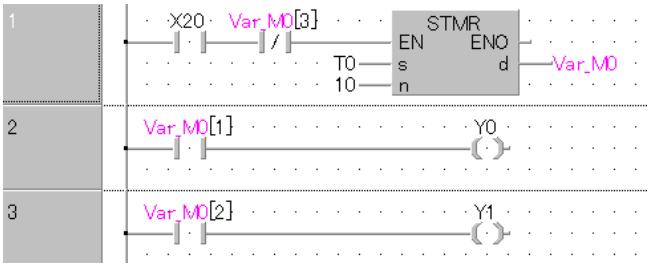
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）　　（出错代码： 4101）

程序示例

以下为将 X20 置为 ON 时，Y0、Y1 每隔 1 秒进行 ON/OFF（闪烁灯）的程序。

（定时器使用 100ms 定时器）

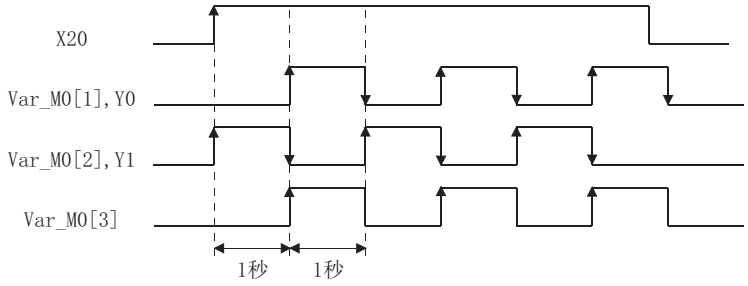
[结构体梯形图]



[ST]

```
STMR(X20 AND NOT(Var_M0[3]),T0,10,Var_M0);
OUT(Var_M0[1],Y0);
OUT(Var_M0[2],Y1);
```

[时序图]



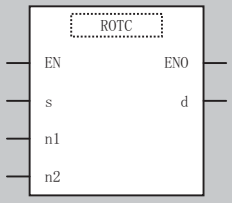
6.8.5 旋转台的就近控制

ROTC

Basic | High performance | Universal | L CPU

ROTC

结构体梯形图



ST

ENO:= ROTC (EN, s, n1, n2, d);

[ ] 中放入下述指令。

ROTC

输入自变量，	EN:	执行条件	:位
	s:	s[0]: 台旋转数测定用	: ANY16 的数组 (0..2)
		s[1]: 调用窗口编号	
		s[2]: 调用物品编号	
	n1:	台分割数 (2 ～ 32767)	: ANY16
	n2:	低速区间数 (0 ～ n1 的值)	: ANY16
输出自变量，	ENO:	执行结果	:位
	d:	d[0]: A 相输入信号	:位的数组 (0..7)
		d[1]: B 相输入信号	
		d[2]: 0 点检测输入信号	
		d[3]: 高速正转输出信号	
		d[4]: 低速正转输出信号	
		d[5]: 停止输出信号	
		d[6]: 低速逆转输出信号	
		d[7]: 高速逆转输出信号	

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-					-
n1	○	○		○					-
n2	○	○		○					-
ⓓ	○	-		-					-

功 能

(1) 在以 n1 中指定的值进行了均等分割的旋转台上，为了将ⓢ[2] 中指定的编号的物品进行取出及放入，在ⓢ[1] 中指定的窗口编号的位置处，对旋转台进行就近旋转控制。

(2) 对于物品编号以及窗口编号，是在逆时针旋转分配的基础上进行控制。

(3) ⓢ[0] 被系统作为计数器使用，用于对 0 号窗口中第几号物品进行计数。
不要通过顺控程序等对其进行数据改写。
如果用户对其进行了改写，将无法正常进行控制。

(4) n2 的设置应小于 n1 中指定的旋转台分割数。

(5) ⓓ[0] 及ⓓ[1] 是用于对旋转台的正转 / 逆转进行检测的 A 相及 B 相输入信号。
旋转方向的判别是通过 A 相为 ON 时的 B 相的上升沿 / 下降沿进行。

- B 相为上升沿时 ： 正转（顺时针旋转）
- B 相为下降沿时 ： 逆转（逆时针旋转）

(6) ⓓ[2] 是指，0 号物品来到 0 号窗口时变为 ON 的 0 点检测信号。
在 ROTC 指令的执行过程中，ⓓ[2] 中指定的软元件变为 ON 时，ⓢ[0] 将被清除。
应预先进行此清除操作之后，再通过 ROTC 指令开始进行就近控制。

(7) ⓓ[3] ～ⓓ[7] 是用于对旋转台的动作进行控制的输出信号。
根据 ROTC 指令的执行结果，ⓓ[3] ～ⓓ[7] 中的某个输出信号将变为 ON。

(8) ROTC 指令的指令变为 OFF 的情况下，不进行就近控制，将ⓓ[3] ～ⓓ[7] 全部置为 OFF。

(9) 在执行的全部程序中，只能使用 1 次 ROTC 指令。
如果使用 2 次及以上，将无法正常执行。

(10) ⓢ[0] ～ⓢ[2] 或 n2 的值大于 n1 的情况下，将变为无处理。

出 错

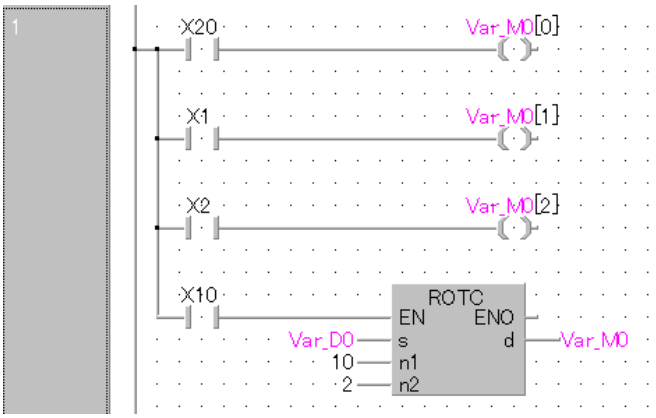
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ或ⓓ中指定的软元件超出了相应软元件的范围时。
（通用型 QCPU、LCPU 时） （出错代码： 4101）

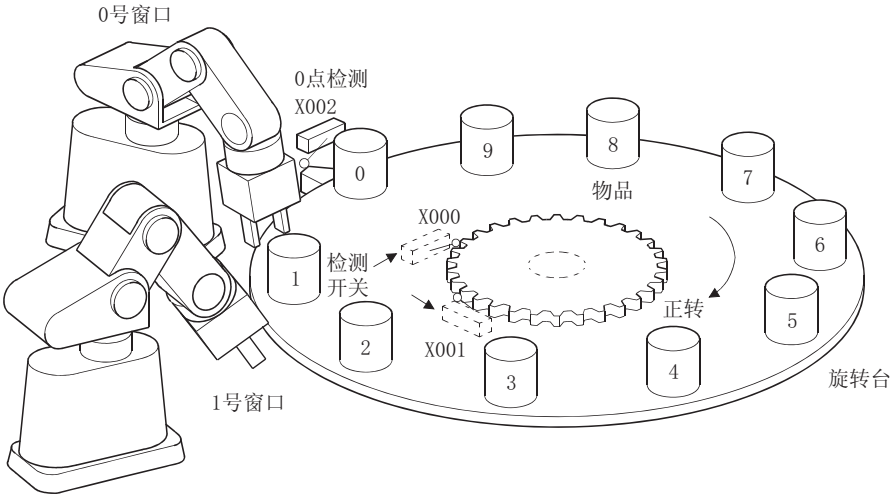
程序示例

以下为将放置于被分割为 10 等份的旋转台上的第 Var_D0[2] 号的物品，在 Var_D0[1] 号窗口中进行取放，在前后 2 区间使旋转台低速旋转的情况下，求出马达的旋转方向及控制速度的程序。

[结构体梯形图]



```
[ST]
OUT(X20,Var_M0[0]);
OUT(X1,Var_M0[1]);
OUT(X2,Var_M0[2]);
ROTC(X10,Var_D0,10,2,Var_M0);
```

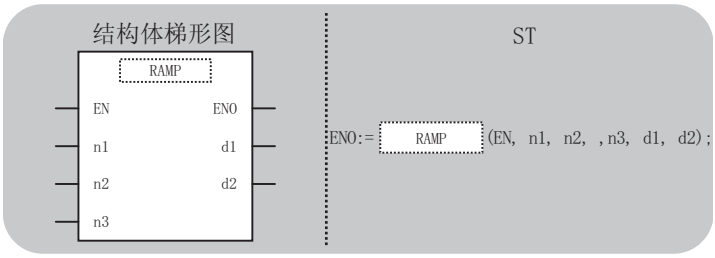


6.8.6 斜坡信号

RAMP

Basic | High performance | Universal | L CPU

RAMP



中放入下述指令。

RAMP

输入自变量， EN: 执行条件 ：位

n1: 初始值 : ANY16

n2: 最终值 : ANY16

n3: 过渡次数 : ANY16

输出自变量， ENO: 执行结果 ：位

d1: d1[0]：当前值 : ANY16 的数组 (0..1)

d1[1]：执行次数

d2: d2[0]：结束软元件 ：位的数组 (0..1)

d2[1]：结束时数据保持选择位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n1	○	○						○	-
n2	○	○						○	-
n3	○	○						○	-
ⓓ1	○	○						-	-
ⓓ2	○	-						-	-

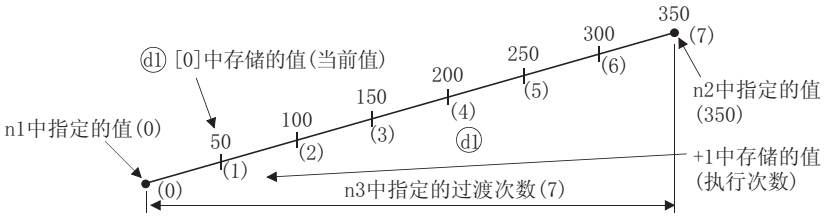
功 能

(1) 执行指令为 ON 时，执行下述处理。

- 从 n1 中指定的值开始至 n2 中指定的值为止，以 n3 中指定的次数进行过渡。
- 在 n3 中，对从 n1 过渡至 n2 的扫描次数（过渡次数）进行指定。
  n3=0 的情况下，变为无处理。
- ⓓ1[1] 为系统所用，用于存储本指令的执行次数。
- 1 次 (1 个扫描）的变化值以下式进行计算。

$$1次的变化值=\frac{(n2中指定的值)-(n1中指定的值)}{(n3中指定的值)}$$

例 以 7 个扫描从 0 变化为 350 时的情况如下所示。



计算 1 次的变化值时，在除不尽的情况下，将进行补偿以通过 n3 中指定的过渡次数使之变为 n2 中指定的值。
因此有时不能变为直线斜坡。

(2) 至最终值为止的过渡结束时，⑫ [0] 中指定的结束软元件将变为 ON。
结束软元件的 ON/OFF 状态以及ⓓ1[0] 的内容取决于⑫ [1] 中指定的软元件的 ON/OFF。

- ⑫ [1] 为 OFF 的情况下，在下一个扫描中将⑫ [0] 置为 OFF 后，RAMP 指令将从初始值开始再次进行过渡。
- ⑫ [1] 为 ON 的情况下，⑫ [0] 将保持 ON 状态不变，ⓓ1[0] 的内容不变化。

(3) 本指令的执行过程中指令变为 OFF 的情况下，ⓓ1[0] 的内容将不再变化。
如果将指令再次置为 ON，RAMP 指令将从初始值开始再次进行过渡。

(4) 在⑫ [0] 中指定的结束软元件为 ON 之前，不要对 n1 及 n2 的值进行变更。
由于每个扫描以同一个计算公式ⓓ1[1] 中存储的值进行计算，因此如果对 n1 及 n2 进行变更，有可能会导致急剧变化。

出 错

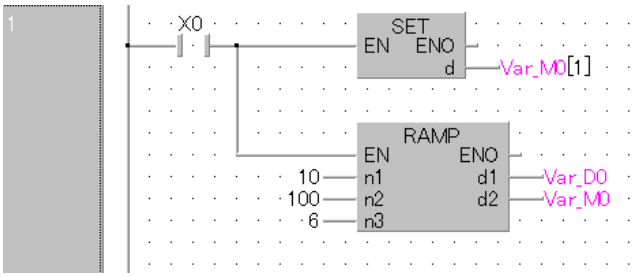
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ1或⑫ 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码 : 4101)

程序示例

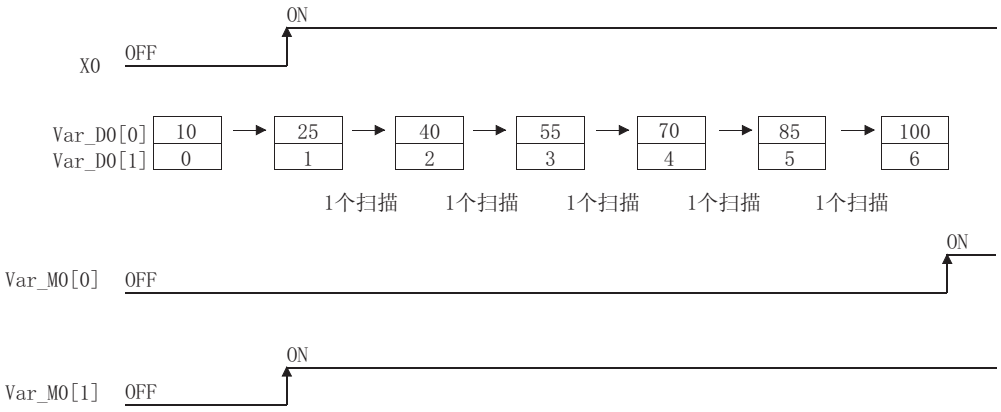
以下为 X0 变为 ON 时，将 Var_D0 的内容通过 6 个扫描从 10 变为 100，过渡结束时将 Var_D0 的内容进行保持的程序。

[结构体梯形图]



```
[ST]
IF X0 THEN
     SET(TRUE,Var_M0[1]);
     RAMP(TRUE,10,100,6,Var_D0,Var_M0);
END_IF;
```

[时序图]



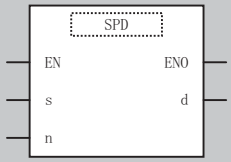
6.8.7 脉冲密度的测定

SPD

Basic | High performance | Universal | L CPU

SPD

结构体梯形图



ST

ENO:= SPD (EN, s, n, d);

[ ] 中放入下述指令。

SPD

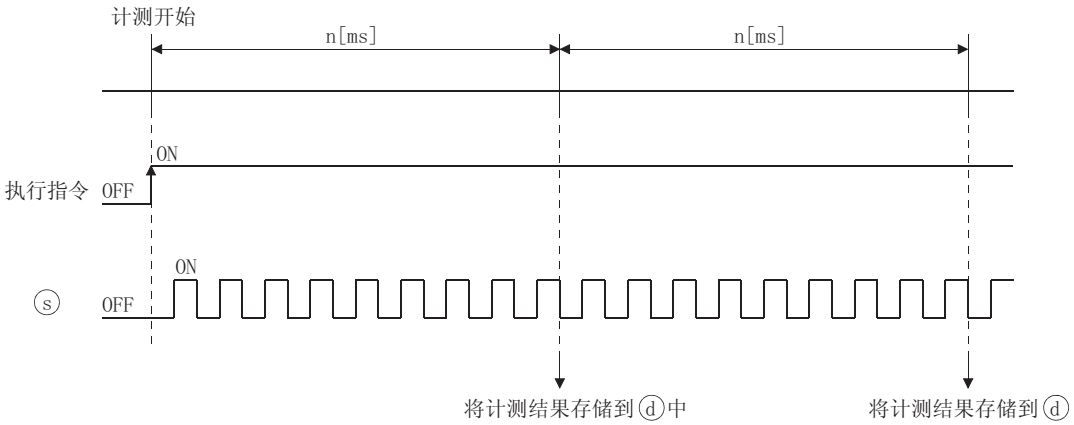
输入自变量， EN: 执行条件 : 位
s: 脉冲输入 : 位
n: 测定时间（单位：ms） : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储测定结果的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○(仅X)	-		-					-
n	△*1	△*1		○					-
ⓓ	-	△*1		-					-

*1 ：不能使用局部软元件及各程序中设置的文件寄存器。

功 能

(1) 将ⓢ中指定的软元件输入在n中指定的时间进行计数，将计数结果存储到ⓓ中指定的软元件中。



(2) 通过 SPD 指令进行的计测结束后，再次从 0 开始进行计测。
使通过 SPD 指令进行的计测中止的情况下，应将执行指令置为 OFF。

要 点

1. 执行 SPD 指令时将自变量软元件的数据登录到 CPU 模块内部的工作区中，实际的计数动作通过系统中断进行。（对于 CPU 模块的工作区中登录的软元件数据，在将执行指令置为 OFF 或 STOP → RUN 时将被清除。）
因此对于可计数的脉冲，其 ON 及 OFF 时间必须大于 CPU 模块的中断间隔时间。
CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、通用型 QCPU、LCPU	1ms

2. 使用高性能型 QCPU 时
n=0 的情况下，将变为无处理。
3. SPD 指令在处于执行状态的全部程序中最多可以使用 6 次。
第 7 次及以后的 UDCNT1 指令将变为无处理。

出 错

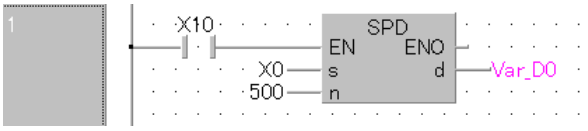
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件超出了相应软元件的范围时。
（通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

以下为将 X10 置为 ON 时，将输入到 X0 中的脉冲以 500ms 为间隔进行计测，将结果存储到 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF X10 THEN
    SPD(TRUE,X0,500,Var_D0);
END_IF;
```

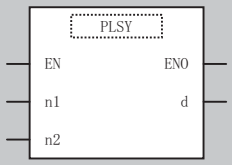
6.8.8 恒定周期脉冲输出

PLSY

Basic | High performance | Universal | L CPU

PLSY

结构体梯形图



ST

```
ENO:=  PLSY  (EN, n1, n2, d);
```

[ ] 中放入下述指令。

PLSY

输入自变量， EN: 执行条件 : 位
n1: 频率或存储软元件编号 : ANY16
n2: 输出次数或存储输出次数的软元件编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 进行脉冲输出的软元件编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n1	○	○							-
n2	○	○							-
ⓓ	r(仅Y)	-							-

功 能

(1) 将 n1 中指定的频率的脉冲以 n2 中指定的次数输出到ⓓ中指定的输出编号 (Y) 的输出模块中。

(2) n1 的频率的可设置范围为 1Hz ～ 100Hz。
n1 超出 1 ～ 100 的范围时，将变为无处理。

(3) n2 的输出次数的可指定范围为 0 ～ 65535(0000H ～ 0FFFFH)。
n2 为 0 的情况下，进行连续的脉冲输出。

(4) 对于ⓓ中指定的脉冲输出，只能指定与输出模块相对应的输出编号。

(5) 在 PLSY 指令的上升沿时开始进行脉冲输出。
PLSY 指令变为 OFF 时，脉冲输出将停止。

要 点

1. 执行 PLSY 指令时，将自变量的软元件数据登录到 CPU 模块的工作区，实际的计数动作是通过系统中断进行处理。
   （对于 CPU 模块的工作区中登录的软元件数据，在将执行指令置为 OFF 或 STOP → RUN 时将被清除。）
   因此对于可计数的脉冲，其 ON 及 OFF 时间必须大于 CPU 模块的中断间隔时间。
   CPU 模块的中断间隔如下所示。

CPU 模块型号	中断间隔
高性能型 QCPU、通用型 QCPU、LCPU	1ms

2. 通过 PLSY 指令进行的脉冲输出过程中（执行指令处于 ON 状态），不要对 PLSY 指令的自变量进行变更。
   对自变量进行变更时，应将执行指令置为 OFF 之后进行。
3. PLSY 指令在处于执行状态的全部程序中最多可以使用 1 次。
   第 2 次及以后的 PLSY 指令将无处理。

出 错

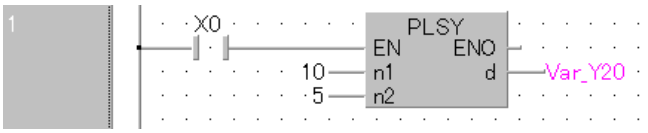
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

以下为将 X0 置为 ON 时，将 10Hz 的脉冲输出到 Var_Y20 中 5 次的程序。

[结构体梯形图]



[ST]

```
PLSY(X0,10,5,Var_Y20);
```

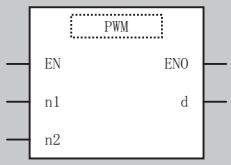
6.8.9 脉冲宽度调制

PWM

Basic | High performance | Universal | L CPU

PWM

结构体梯形图



ST

ENO:= PWM (EN, n1, n2, d);

中放入下述指令。

PWM

输入自变量， EN: 执行条件 :位

n1: ON 时间或存储 ON 时间的软元件编号 : ANY16

n2: 周期或存储周期的软元件编号 : ANY16

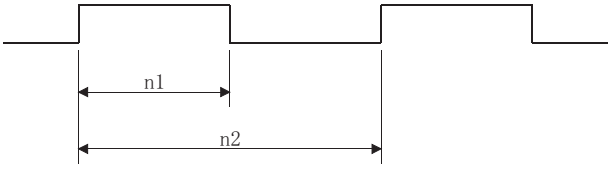
输出自变量， ENO: 执行结果 :位

d: 进行脉冲输出的软元件编号 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n1	○	○							-
n2	○	○							-
ⓓ	△(仅Y)	-							-

功 能

(1) 将 n1 中指定的 ON 时间及 n2 中指定的周期的脉冲输出到ⓓ中指定的输出模块中。



(2) n1、n2 的设置时间如下所示。

CPU 模块型号	n1、n2 的设置范围 [ms]*1
高性能型 QCPU、通用型 QCPU、LCPU	1 ～ 65535(0001 ～ 0FFFF)

*1 : n1 中指定的值应小于 n2 中指定的值。

要 点

1. 执行 PWM 指令时，将自变量的软元件数据登录到 CPU 模块的工作区，实际的计数动作是通过系统中断进行处理。
   （对于 CPU 模块的工作区中登录的软元件数据，在将执行指令置为 OFF 或 STOP → RUN 时将被清除。）
   CPU 模块的中断间隔如下所示。

CPU 模块型号	n1、n2 的中断间隔
高性能型 QCPU、通用型 QCPU、LCPU	1ms

   因此 PWM 指令在处于执行状态的全部程序中最多可以使用 1 次。
2. 在下述情况下将变为无处理。
   • n1,n2 为 0 时
   • n2≧n1 时
3. 在通过 PWM 指令进行的脉冲输出过程中（执行指令为 ON 状态），不要对 PWM 指令的自变量进行变更。
   如果要对自变量进行变更，应将执行指令置为 OFF 之后再进行变更。

出 错

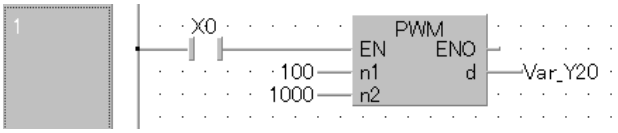
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

以下为将 X0 置为 ON 时，将 100ms 的脉冲每隔 1 秒输出到 Var_Y20 中的程序。

[结构体梯形图]



[ST]

```
PWM(X0,100,1000,Var_Y20);
```

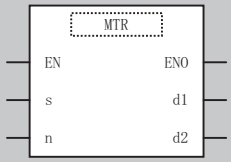
6.8.10 矩阵输入

MTR

Basic | High performance | Universal | L CPU

MTR

结构体梯形图



ST

ENO:= MTR (EN, s, n, d1, d2);

□ 中放入下述指令。

MTR

输入自变量， EN: 执行条件 : 位
s: 输入的起始软元件 : 位
n: 输入列数 : ANY16
输出自变量， ENO: 执行结果 : 位
d1: 输出的起始软元件 : 位
d2: 存储矩阵输入数据的软元件的起始编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○(仅X)		-						-
n	○		○						-
ⓓ1	○(仅Y)		-						-
ⓓ2	○		-						-

功 能

(1) 对ⓢ中指定的输入编号以后连接的 16 点× n 列的输入依次进行读取，将读取的输入数据存储到ⓓ2 中指定的软元件以后。

(2) 进行读取时，在 1 个扫描中读取 1 列 (16 点)。

(3) 依次反复对第 1 列至第 n 列进行读取。

(4) 在ⓓ2 中指定的软元件的后面，在起始的 16 点中存储第 1 列的数据，在下一个 16 点中存储第 2 列的数据。
因此在ⓓ2 中指定的软元件算起的 16 × n 点被 MTR 指令所占用。

(5) ⓓ1是用于选择要读取的列的输出，由系统自动地进行 ON/OFF。
使用ⓓ1中指定的软元件算起的 n 点。

(6) 在ⓢ， ⓓ1， ⓓ2 中，只能指定 16 的倍数的软元件编号。

(7) 在 n 中的可指定范围为 2 ～ 8。

(8) 在下述情况下将变为无处理。

- 在ⓢ, ⓓ1, ⓓ2 中指定的软元件 No. 不是 16 的倍数时。
- ⓢ中指定的软元件超出了实际输入的范围时。
- ⓓ1中指定的软元件超出了实际输出的范围时。
- ⓓ2 中指定的软元件以后的 16 × n 点超出了相应软元件的范围时。
- 在 n 中的指定超出了 2 ～ 8 的范围时。

出 错

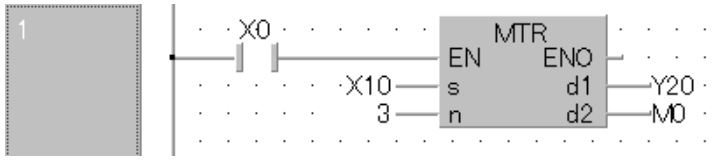
在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定了除输入 (X) 以外时。 (出错代码： 4101)
- ⓓ1中指定了除输出 (Y) 以外时。 (出错代码： 4101)

程序示例

以下为将 X0 置为 ON 时，对 X10 以后连接的 16 点× 3 列的矩阵进行读取后，存储到 M0 以后的程序。

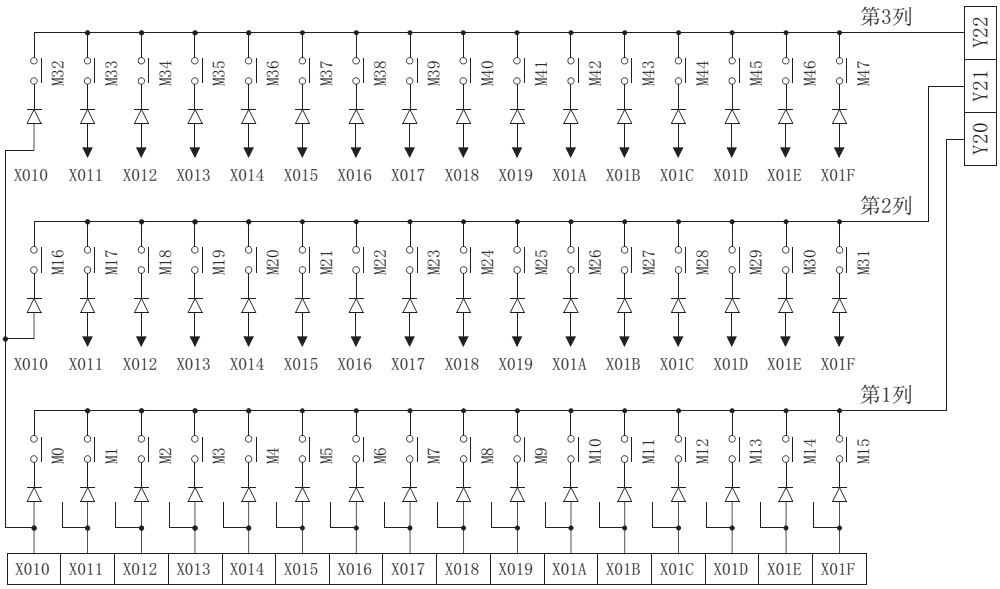
[结构体梯形图]



[ST]

```
MTR(X0,X10,3,Y20,M0);
```

[动作]



注意事项

(1) MTR 指令对实际输入输出进行直接操作，因此应加以注意。

即使 MTR 的指令变为 OFF，通过 MTR 指令置为 ON 的输出ⓓⓛ也不会变为 OFF。

应通过顺控程序将ⓓⓛ中指定的输出置为 OFF。

(2) 对于 MTR 的执行间隔，应设置为长于输入模块与输出模块的响应时间的合计值。

如果 MTR 的执行间隔短于上述时间，将无法正常地对输入进行获取。
在顺控程序的扫描时间较短的情况下，应在恒定扫描中将扫描时间设置为长于响应时间的合计值。

7 应用指令

7.1 逻辑运算指令

执行逻辑运算指令时将逻辑和、逻辑积等的逻辑运算以 1 位为单位进行。

分类	处理内容	运算公式	示例		
			A	B	Y
逻辑积 (AND)	仅在输入 A、B 均为 1 时才变为 1，除此以外将变为 0。	Y=A・B	0	0	0
			0	1	0
			1	0	0
			1	1	1
逻辑和 (OR)	仅在输入 A、B 均为 0 时才变为 0，除此以外将变为 1。	Y=A+B	0	0	0
			0	1	1
			1	0	1
			1	1	1
排他逻辑和 (XOR)	输入 A 与 B 相等时变为 0，不相等时变为 1。	$Y=\overline{A}・B+A・\overline{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
否定排他逻辑和 (XNR)	输入 A 与 B 相等时变为 1，不相等时变为 0。	$Y=(\overline{A}+B)(A+\overline{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

7.1.1 16 位 /32 位数据逻辑积

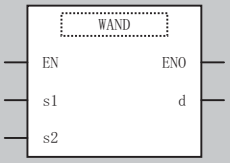
WAND, DAND

Basic | High performance | Universal | L CPU

WAND(P)
DAND(P)

P: 执行条件 : ↵

结构体梯形图



ST

ENO:= WAND (EN, s1, s2, d);

[ ] 中放入下述指令。

WAND	WANDP
DAND	DANDP

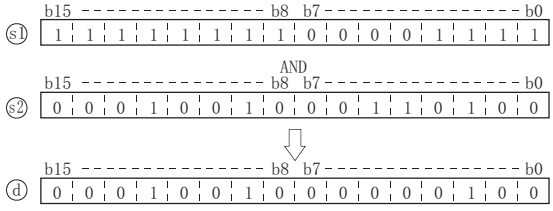
输入自变量， EN: 执行条件 : 位
s1: 进行逻辑积的数据 : ANY16/32
s2: 进行逻辑积的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 逻辑积的结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1	○							○	-
ⓢ2	○							○	-
ⓓ	○							-	-

功 能

WAND(P)

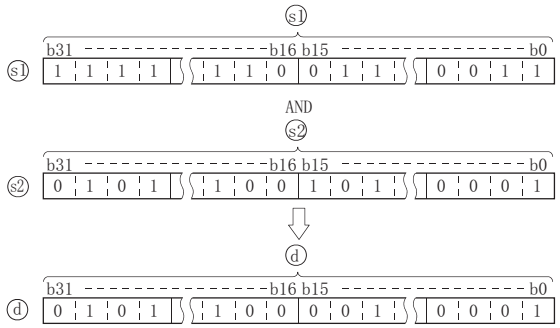
(1) 将ⓢ1 中指定的软元件的 16 位数据与ⓢ2 中指定的软元件的 16 位数据逐位进行逻辑积后，将结果存储到ⓓ 中指定的软元件中。



(2) 位软元件的情况下，将位数指定的点数以后的位软元件作为 0 进行运算。
（参阅程序示例 (1), (2)）

DAND(P)

(1) 将 ⓢ1 中指定的软元件的 32 位数据与 ⓢ2 中指定的软元件的 32 位数据逐位进行逻辑积运算后，将结果存储到 ⓓ 中指定的软元件中。



(2) 位软元件的情况下，将位数指定的点数以后的位软元件作为 0 进行运算。
（参阅程序示例 (3)）

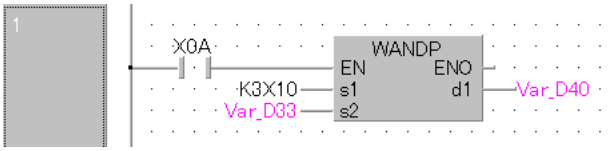
出 错

不存在 WAND(P)、DAND(P) 指令相关的运算出错。

程序示例

(1) 以下为 X0A 变为 ON 时，将 X10 ～ X1B 的数据与 Var_D33 进行逻辑积运算后，将其结果存储到 Var_D40 中的程序。

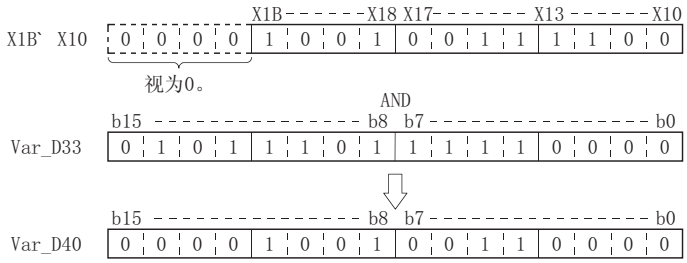
[结构体梯形图]



[ST]

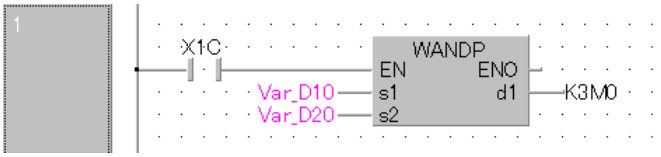
```
WANDP(X0A,K3X10,Var_D33,Var_D40);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将 Var_D10 与 Var_D20 进行逻辑积运算后，将结果存储到 M0 ～ M11 中的程序。

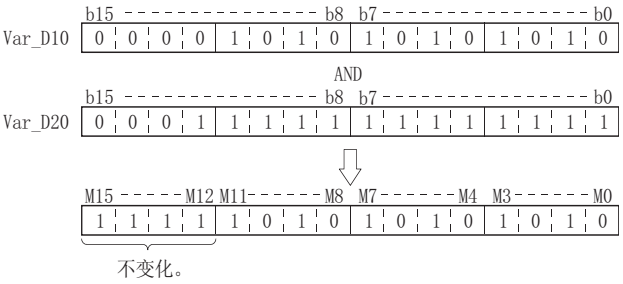
[结构体梯形图]



[ST]

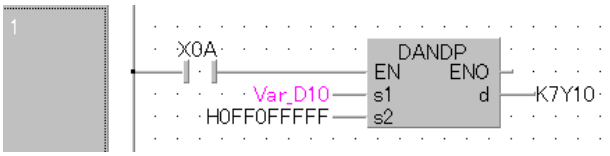
```
WANDP(X1C,Var_D10,Var_D20,K3M0);
```

[动作]



(3) 以下为 X0A 变为 ON 时，将 Var_D10 的 BCD8 位中的 100000 的位（低位算起的第 6 位）屏蔽为 0 后，将运算结果输出到 Y10 ～ Y2B 中的程序。

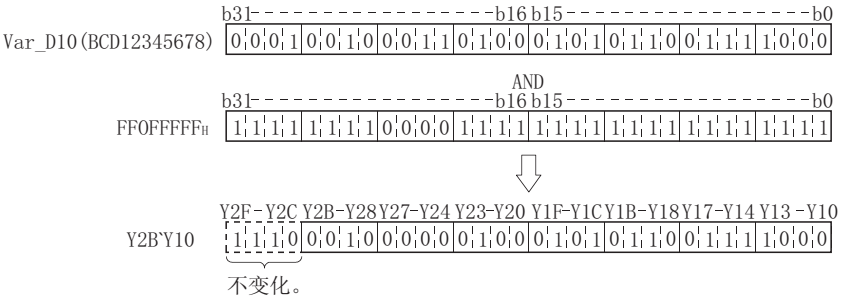
[结构体梯形图]



[ST]

```
DANDP(X0A,Var_D10,H0FF0FFFFF,K7Y10);
```

[动作]



7.1.2 块逻辑积

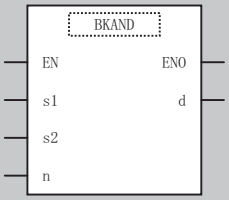
BKAND

Basic | High performance | Universal | L CPU

BKAND(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= BKAND (EN, s1, s2 , n, d);
```

[　　] 中放入下述指令。

BKAND　　BKANDP

输入自变量，EN：　执行条件　：位

s1：　存储进行逻辑积的数据的软元件的起始编号　：ANY16

s2：　进行逻辑积的数据或存储数据的软元件的起始编号　：ANY16

n：　运算数据个数　：ANY16

输出自变量，ENO：　执行结果　：位

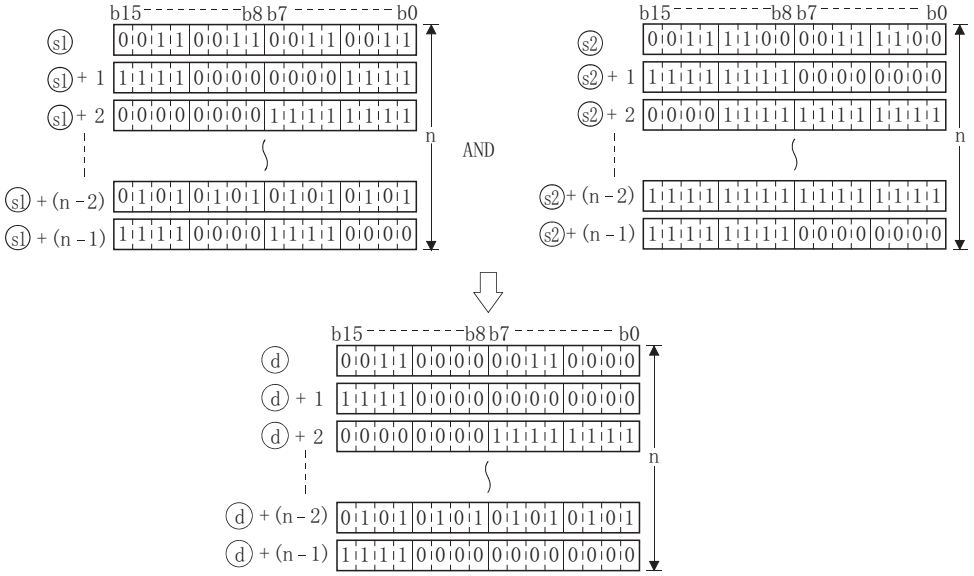
d：　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1 *1	－	○		－				－	－
ⓢ2 *1	－	○		－				○	－
n	○	○		○				○	－
ⓓ *1	－	○		－				－	－

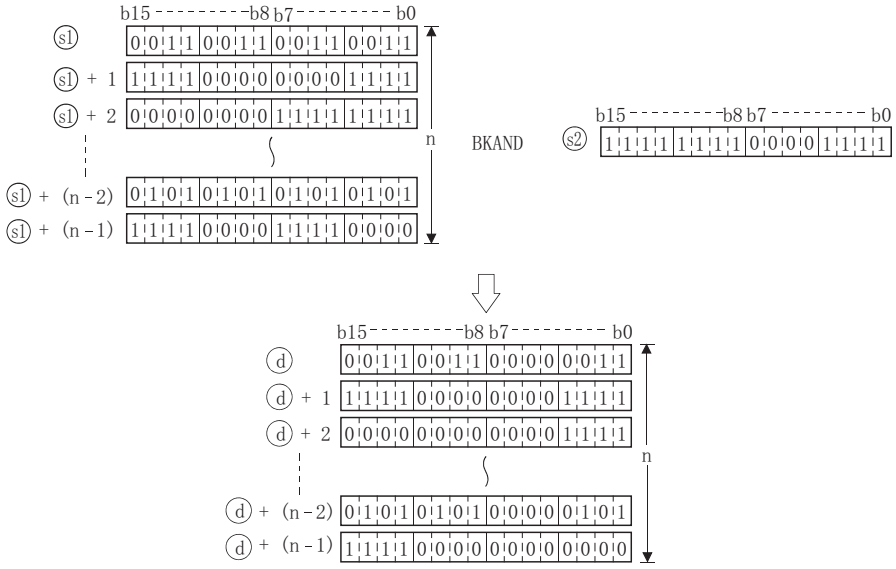
*1 ：ⓢ1 与ⓓ或ⓢ2 与ⓓ可以指定为相同的软元件编号。

功 能

(1) 将㉑中指定的软元件算起 n 点的内容与㉒中指定的软元件算起 n 点的内容进行逻辑积运算后，将结果存储到ⓓ中指定的软元件以后。



(2) ㉒中可指定的常数范围为 -32768 ～ 32767(BIN 16 位)。



出 错

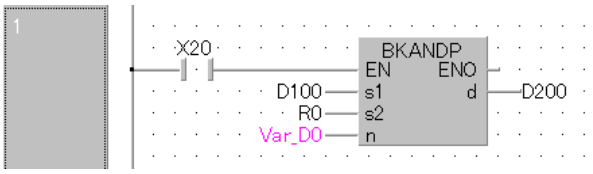
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ㉑，㉒，ⓓ的软元件算起 n 点的范围超出了相应软元件时。
  (出错代码：4101)
- ㉑算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉑与ⓓ中指定了同一软元件时除外。) (出错代码：4101)
- ㉒算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉒与ⓓ中指定了同一软元件时除外。) (出错代码：4101)

程序示例

以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的数据与 R0 ～ R2 中存储的值的数据进行逻辑积运算后，将其结果存储到 D200 以后的程序。

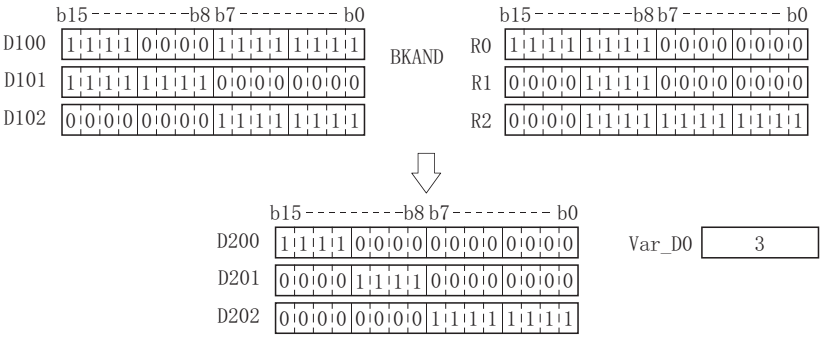
[结构体梯形图]



[ST]

```
D0:=K3;
BKANDP(X20,D100,R0,Var_D0,D200);
```

[动作]



7.1.3 16 位 /32 位数据逻辑和

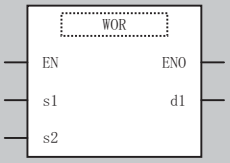
WOR, DOR

Basic | High performance | Universal | L CPU

WOR (P)
DOR (P)

P: 执行条件 : ⏎

结构体梯形图



ST

ENO:= WOR (EN, s1, s2, d1);

中放入下述指令。

WOR	WORP
DOR	DORP

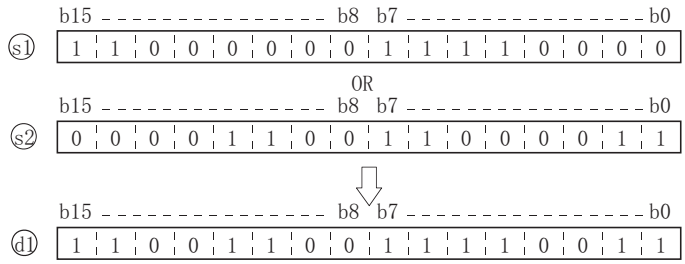
输入自变量， EN: 执行条件 : 位
s1: 进行逻辑和的数据 : ANY16/32
s2: 进行逻辑和的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d1: 逻辑和的结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1		○						○	-
ⓢ2		○						○	-
ⓓ1		○						-	-

功 能

WOR(P)

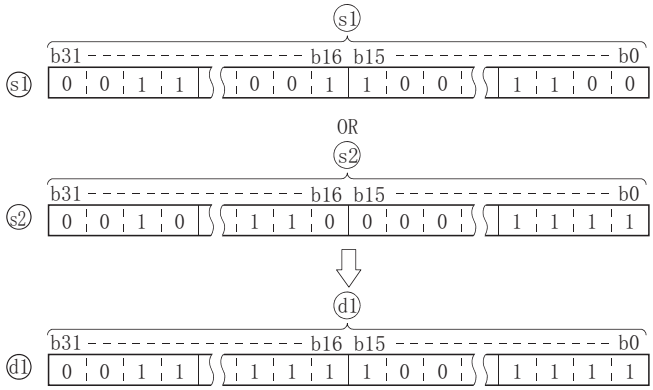
(1) 将ⓢ1 中指定的软元件的 16 位数据与ⓢ2 中指定的软元件的 16 位数据逐位进行逻辑和运算后，将结果存储到ⓓ 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。
（参阅程序示例 (1)）

DOR(P)

(1) 将 ⓢ1 中指定的软元件的 32 位数据与 ⓢ2 中指定的软元件的 32 位数据逐位进行逻辑和运算后，将结果存储到 ⓓ1 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。
（参阅程序示例 (2)）

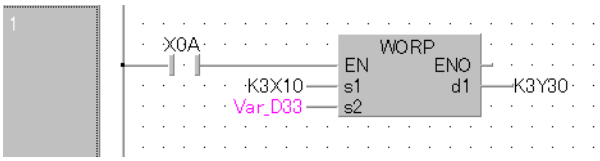
出 错

不存在 WOR(P)、DOR(P) 指令相关的运算出错。

程序示例

(1) 以下位 X0A 变为 ON 时，将 X10 ～ X1B 的数据与 Var_D33 的数据进行逻辑和运算后，将其结果输出到 Y30 ～ Y3B 中的程序。

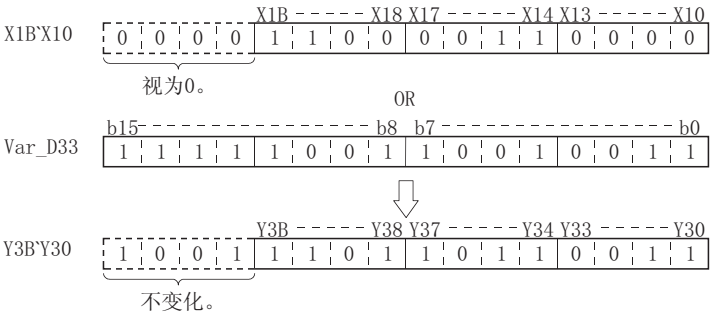
[结构体梯形图]



[ST]

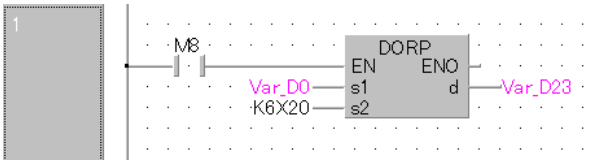
```
WORP(X0A,K3X10,Var_D33,K3Y30);
```

[动作]



(2) 以下为 M8 变为 ON 时，将 Var_D0 的 32 位数据与 X20 ～ X37 的 24 位数据进行逻辑和运算后，将结果存储到 Var_D23 中的程序。

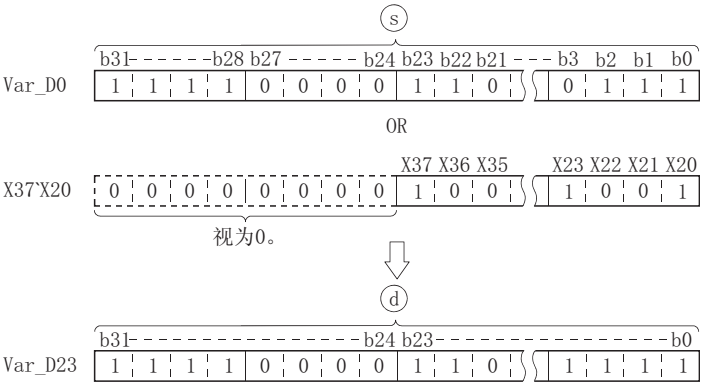
[结构体梯形图]



[ST]

```
DORP(M8,Var_D0,K6X20,Var_D23);
```

[动作]



7.1.4 块逻辑和

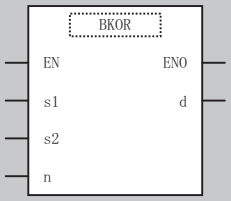
BKOR

Basic | High performance | Universal | L CPU

BKOR(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= BKOR (EN, s1, s2, n, d);
```

□ 中放入下述指令。

BKOR　　BKORP

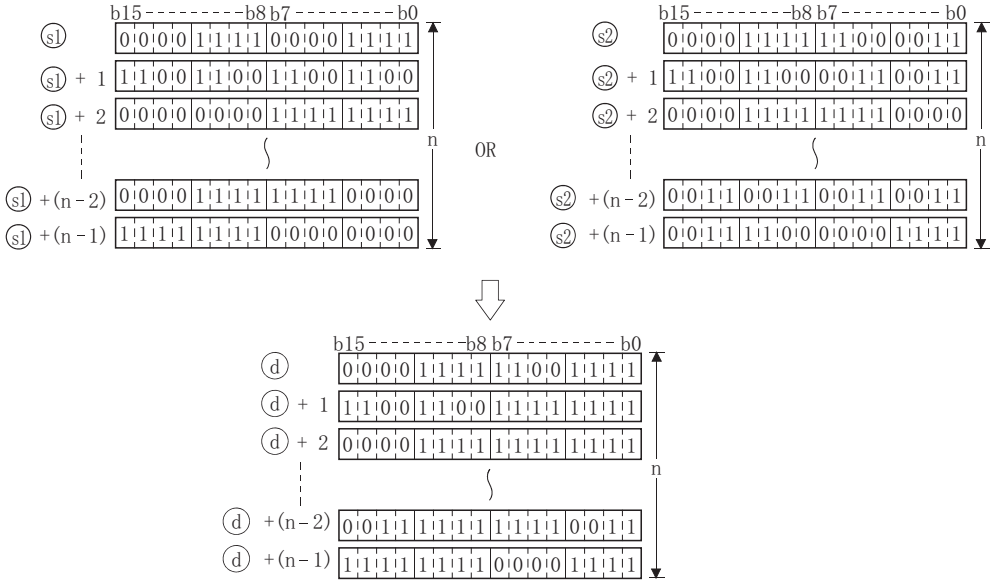
输入自变量，EN：　执行条件　：位
　　s1：　存储进行逻辑和的数据的软元件的起始编号　：ANY16
　　s2：　进行逻辑和的数据或存储数据的软元件的起始编号　：ANY16
　　n：　运算数据个数　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储逻辑和的运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1 *1	－	○		－				－	－
ⓢ2 *1	－	○		－				○	－
n	○	○		○				○	－
ⓓ *1	－	○		－				－	－

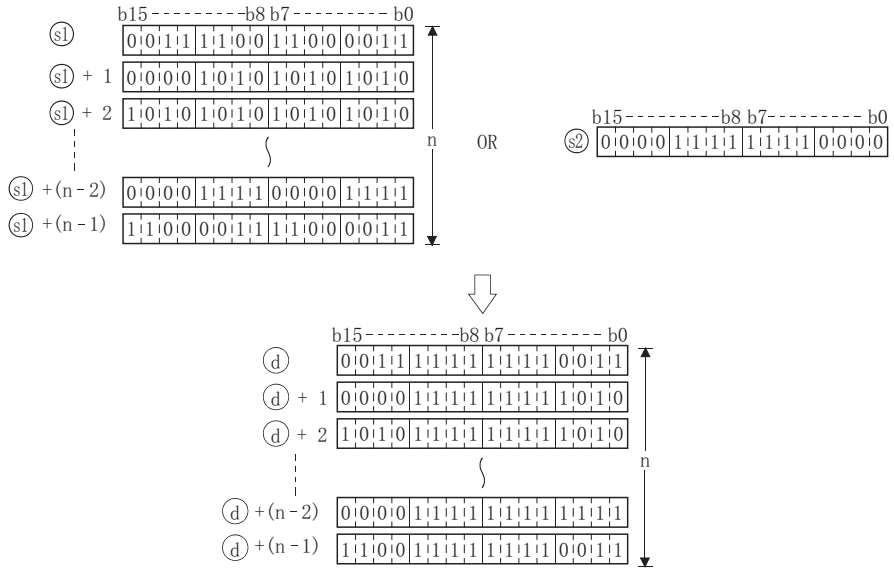
*1 ：ⓢ1与ⓓ或ⓢ2与ⓓ可以指定为相同的软元件编号。

功 能

(1) 将㉑中指定的软元件算起 n 点的内容与㉒中指定的软元件算起 n 点的内容进行逻辑和运算后，将结果存储到ⓓ中指定的软元件以后。



(2) ㉒中可指定的常数范围为 -32768 ～ 32767(BIN 16 位)。



出 错

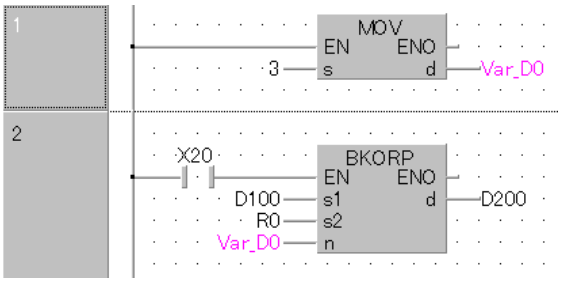
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ㉑，㉒，ⓓ的软元件算起 n 点的范围超出了相应软元件时。 (出错代码： 4101)
- ㉑算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉑与ⓓ中指定了同一软元件时除外。) (出错代码： 4101)
- ㉒算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉒与ⓓ中指定了同一软元件时除外。) (出错代码： 4101)

程序示例

以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的数据与 R0 ～ R2 中存储的值的数据进行逻辑和运算后，将其结果存储到 D200 以后的程序。

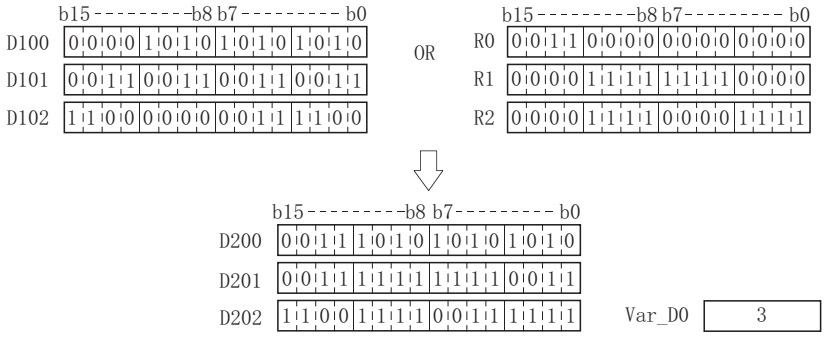
[结构体梯形图]



[ST]

```
Var_D0:=3;
BKORP(X20,D100,R0,Var_D0,D200);
```

[动作]



7.1.5 16 位 /32 位数据排他逻辑和

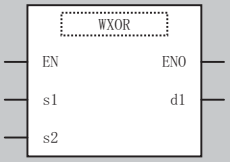
WXOR, DXOR

Basic | High performance | Universal | L CPU

WXOR(P)
DXOR(P)

P: 执行条件 : ↵

结构体梯形图



ST

ENO:= WXOR (EN, s1, s2, d);

□ 中放入下述指令。

WXOR	WXORP
DXOR	DXORP

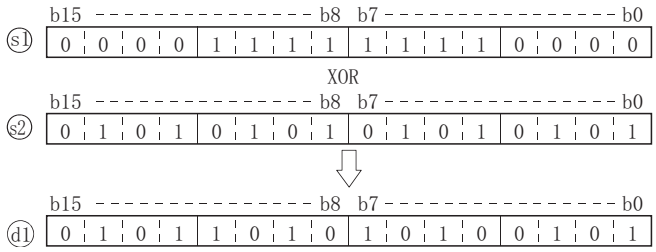
输入自变量， EN: 执行条件 : 位
s1: 进行排他逻辑和的数据 : ANY16/32
s2: 进行排他逻辑和的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d1: 排他逻辑和的结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1		○						○	-
ⓢ2		○						○	-
ⓓ1		○						-	-

功 能

WXOR(P)

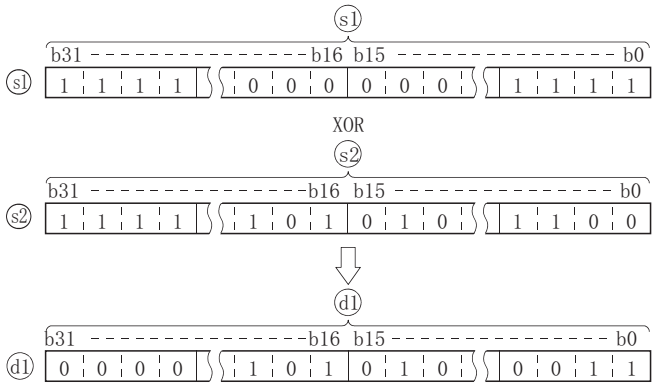
(1) 将ⓢ1 中指定的软元件的 16 位数据与ⓢ2 中指定的软元件的 16 位数据逐位进行排他逻辑和运算后，将结果存储到ⓓ1 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。
（参阅程序示例 (1)）

DXOR(P)

(1) 将 ⓢ1 中指定的软元件的 32 位数据与 ⓢ2 中指定的软元件的 32 位数据逐位进行排他逻辑和运算后，将结果存储到 ⓓ1 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。

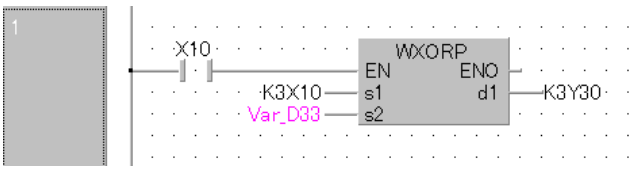
出 错

不存在 WXOR(P)、DXOR(P) 指令相关的运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，将 X10 ～ X1B 的数据与 Var_D33 的数据进行排他逻辑和运算后，将其结果输出到 Y30 ～ Y3B 中的程序。

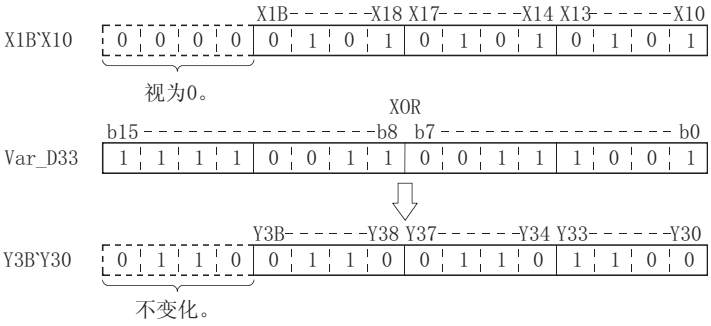
[结构体梯形图]



[ST]

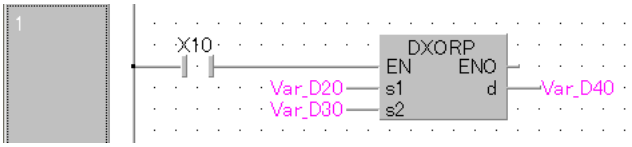
```
WXORP(X10,K3X10,Var_D33,K3Y30);
```

[动作]



(2) 以下为 X10 变为 ON 时，将 Var_D20 的数据与 Var_D30 的数据进行排他逻辑和运算后，将其结果存储到 Var_D40 中的程序。

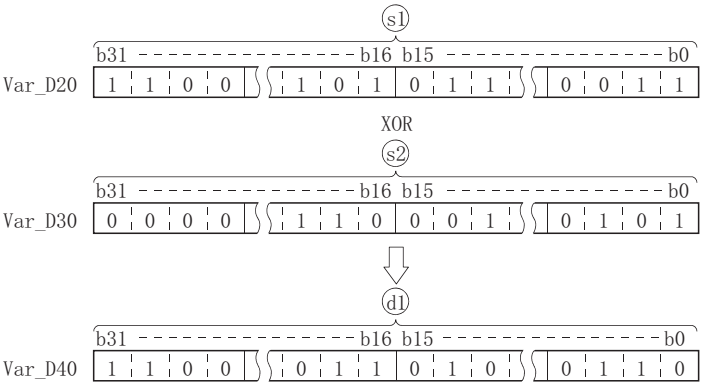
[结构体梯形图]



[ST]

```
DXORP(X10,Var_D20,Var_D30,Var_D40);
```

[动作]



7.1.6 块排他逻辑和

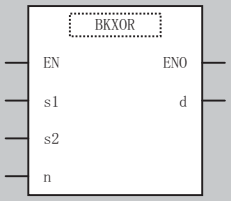
BKXOR

Basic | High performance | Universal | L CPU

BKXOR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BKXOR (EN, s1, s2, n, d);

□ 中放入下述指令。

BKXOR　　BKXORP

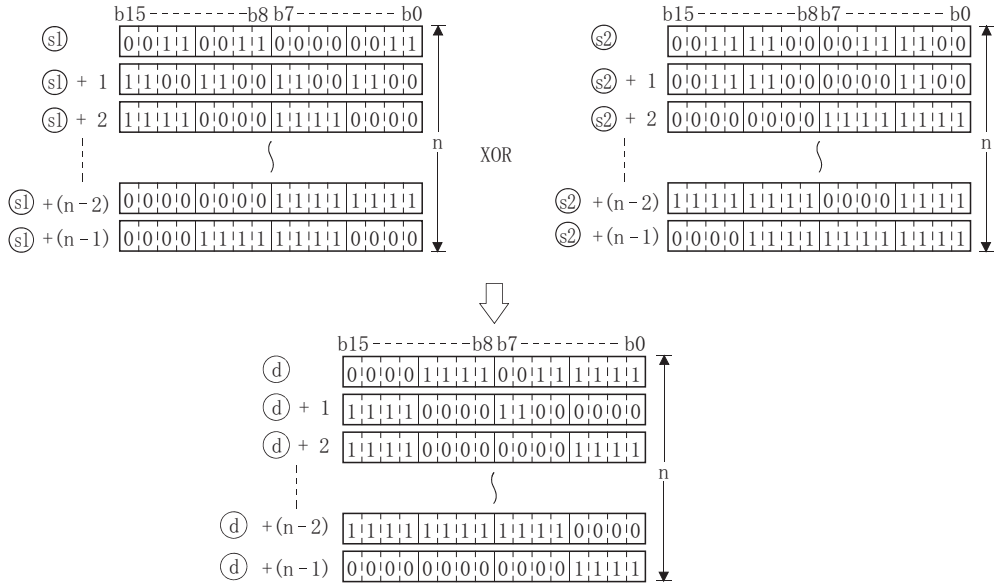
输入自变量，EN:　执行条件　：位
　　s1:　存储进行排他逻辑和的数据的软元件的起始编号　：ANY16
　　s2:　进行排他逻辑和的数据或存储数据的软元件的起始编号　：ANY16
　　n:　运算数据个数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1 *1	－	○		－				－	－
ⓢ2 *1	－	○		－				○	－
n	○	○		○				○	－
ⓓ *1	－	○		－				－	－

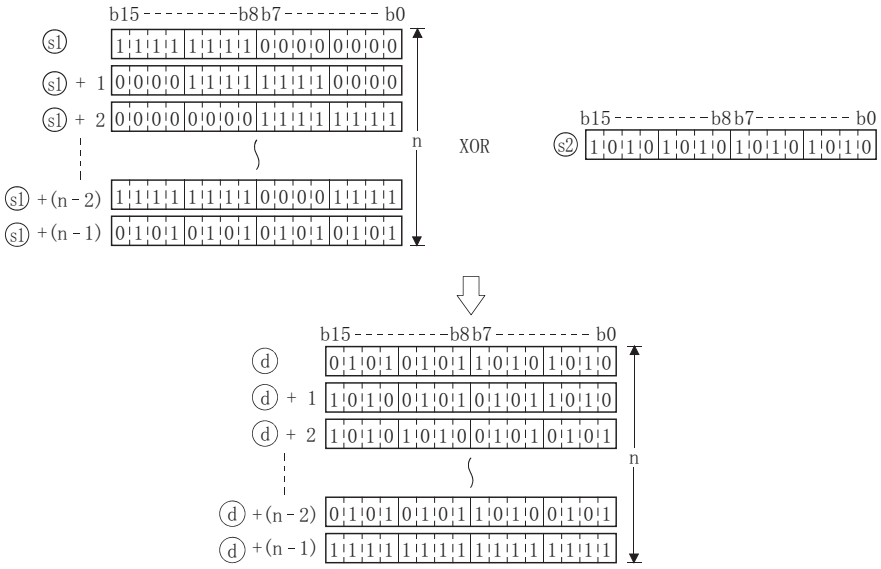
*1 ：ⓢ1 与ⓓ 或ⓢ2 与ⓓ 可以指定为相同的软元件编号。

功 能

(1) 将㉑中指定的软元件算起 n 点的内容与㉒中指定的软元件算起 n 点的内容进行排他逻辑和运算后，将结果存储到ⓓ中指定的软元件以后。



(2) ㉒中可指定的常数范围为 -32768 ～ 32767(BIN 16 位)。



出 错

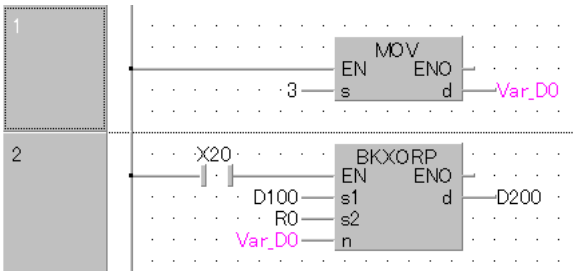
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ㉑，㉒，ⓓ的软元件算起 n 点的范围超出了相应软元件时。 (出错代码： 4101)
- ㉑算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉑与ⓓ中指定了同一软元件时除外。) (出错代码： 4101)
- ㉒算起 n 点的软元件范围与ⓓ算起 n 点的软元件范围有部分重叠时。
  (㉒与ⓓ中指定了同一软元件时除外。) (出错代码： 4101)

程序示例

以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的数据与 R0 ～ R2 中存储的值的数据进行排他逻辑和运算后，将其结果存储到 D200 以后的程序。

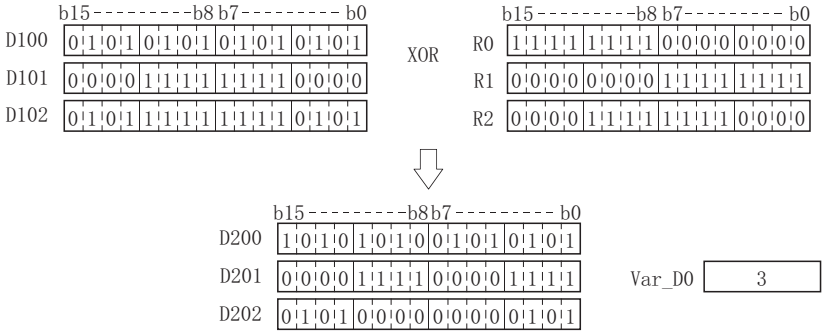
[结构体梯形图]



[ST]

```
Var_D0:=3;
BKXORP(X20,D100,R0,Var_D0,D200);
```

[动作]



7.1.7 16 位 /32 位数据否定排他逻辑和

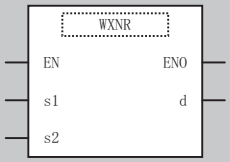
WXNR, DXNR

Basic | High performance | Universal | L CPU

WXNR(P)
DXNR(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= WXNR (EN, s1, s2, d);

中放入下述指令。

WXNR	WXNRP
DXNR	DXNRP

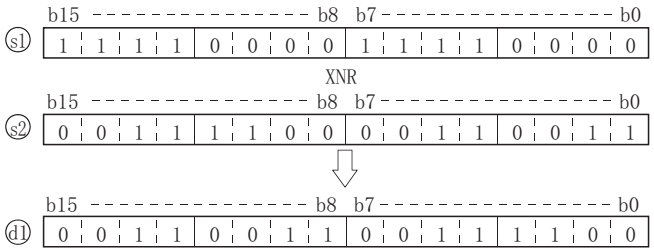
输入自变量， EN: 执行条件 : 位
s1: 进行否定排他逻辑和的数据 : ANY16/32
s2: 进行否定排他逻辑和的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d1: 否定排他逻辑和的结果 : ANY16/32

设置数据	内部软元件 位	内部软元件 字	R, ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K,H	其它
ⓢ1				○				○	-
ⓢ2				○				○	-
ⓓ1				○				-	-

功 能

WXNR(P)

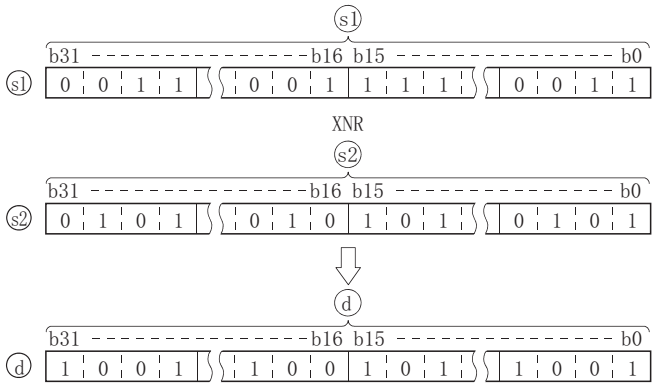
(1) 将ⓢ1 中指定的软元件的 16 位数据与ⓢ2 中指定的软元件的 16 位数据进行否定排他逻辑和运算后，将结果存储到ⓓ1 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。

DXNR(P)

(1) 将 ⓢ1 中指定的软元件的 32 位数据与 ⓢ2 中指定的软元件的 32 位数据进行否定排他逻辑和运算后，将结果存储到 ⓓ 中指定的软元件中。



(2) 位软元件的情况下，位数指定的点数以后的位软元件将被作为 0 进行运算。

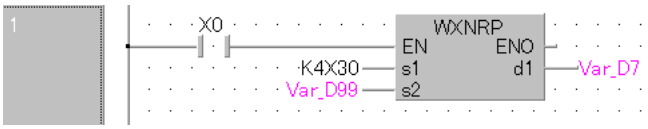
出 错

不存在 WXNR(P)、DXNR(P) 指令相关的运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，将 X30 ～ X3F 的 16 位数据与 Var_D99 的数据进行否定排他逻辑和运算后，将其结果存储到 Var_D7 中的程序。

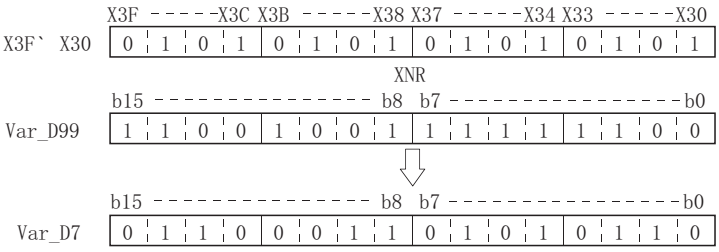
[结构体梯形图]



[ST]

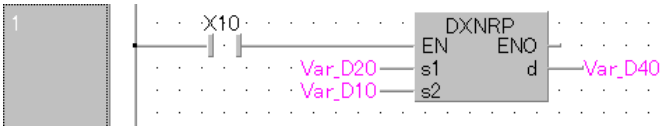
```
WXNRP(X0,K4X30,Var_D99,Var_D7);
```

[动作]



(2) 以下为 X10 变为 ON 时，将 Var_D20 的 32 位数据与 Var_D10 的数据进行否定排他逻辑和运算后，将其结果存储到 Var_D40 中的程序。

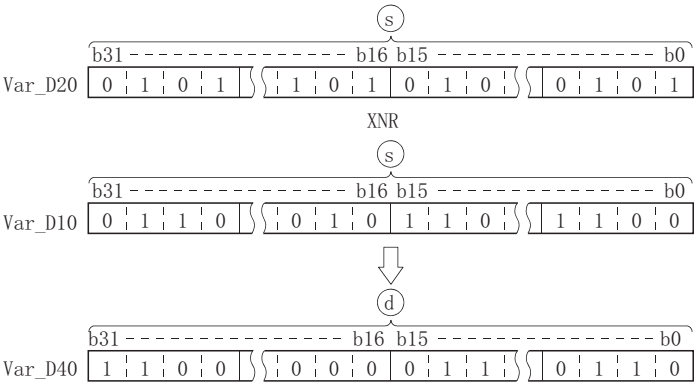
[结构体梯形图]



[ST]

```
DXNRP(X10,Var_D20,Var_D10,Var_D40);
```

[动作]



7.1.8 块否定排他逻辑和

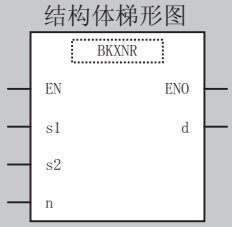
BKXNR

Basic | High performance | Universal | L CPU

BKXNR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BKXNR (EN, s1, s2, n, d);

[　　] 中放入下述指令。

BKXNR　　BKXNRP

输入自变量，EN:　执行条件　：位

s1:　存储进行否定排他逻辑和的数据的软元件的起始编号　：ANY16

s2:　进行否定排他逻辑和的数据或存储数据的软元件的起始编号　：ANY16

n:　运算数据个数　：ANY16

输出自变量，ENO:　执行结果　：位

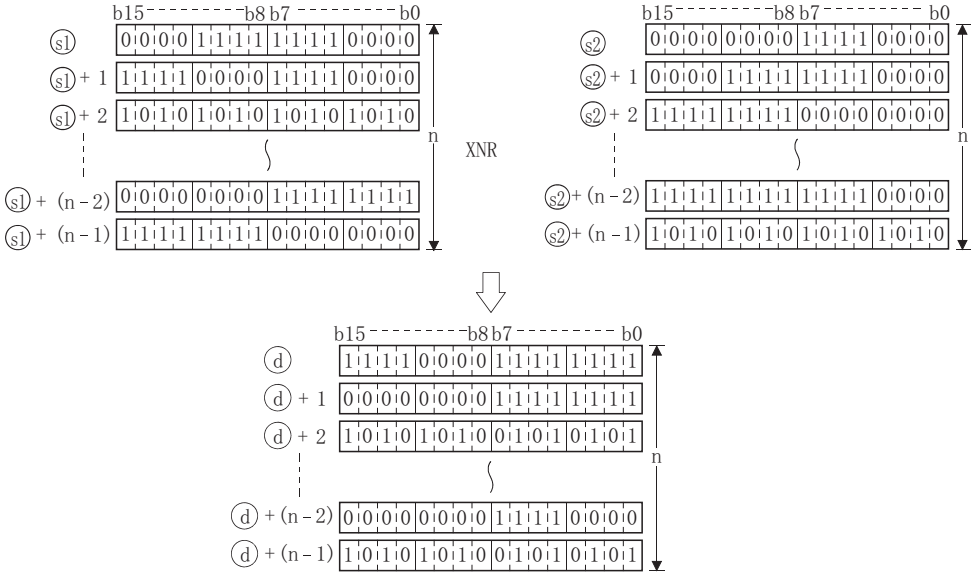
d:　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
ⓢ1 *1	－	○		－				－	－
ⓢ2 *1	－	○		－				○	－
n	○	○		○				○	－
ⓓ *1	－	○		－				－	－

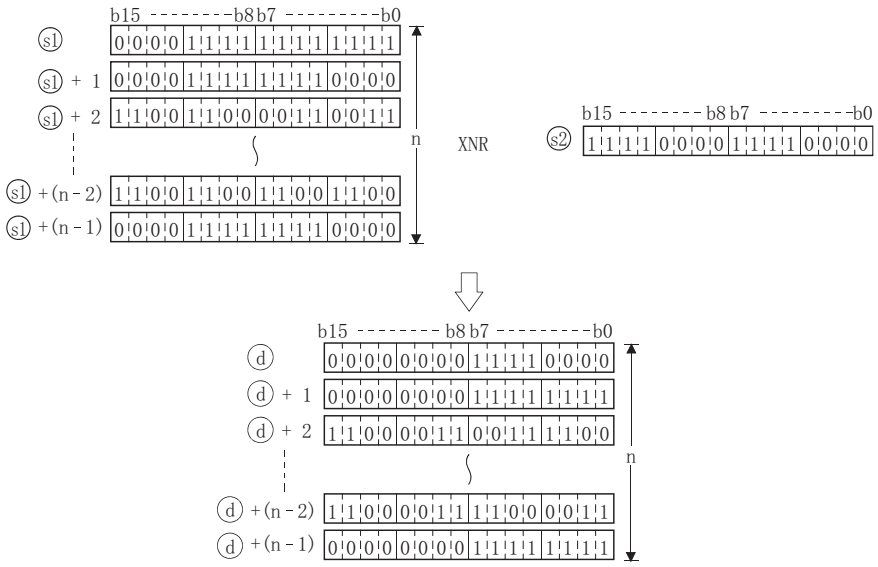
*1 :ⓢ1与ⓓ或ⓢ2与ⓓ可以指定为相同的软元件编号。

功 能

(1) 将 ⓢ1 中指定的软元件算起 n 点的内容与 ⓢ2 中指定的软元件算起 n 点的内容进行否定排他逻辑和运算后，将结果存储到 ⓓ 中指定的软元件以后。



(2) ⓢ2 中可指定的常数范围为 -32768 ～ 32767(BIN 16 位)。



出 错

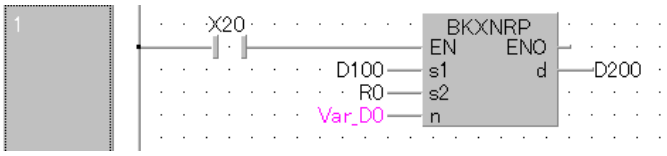
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1，ⓢ2，ⓓ 的软元件算起 n 点的范围超出了相应软元件时。 (出错代码： 4101)
- ⓢ1 算起 n 点的软元件范围与 ⓓ 算起 n 点的软元件范围有部分重叠时。
  (ⓢ1 与 ⓓ 中指定了同一软元件时除外。) (出错代码： 4101)
- ⓢ2 算起 n 点的软元件范围与 ⓓ 算起 n 点的软元件范围有部分重叠时。
  (ⓢ2 与 ⓓ 中指定了同一软元件时除外。) (出错代码： 4101)

程序示例

以下为 X20 变为 ON 时，将 D100 ～ D102 中存储的值的数据与 R0 ～ R2 中存储的值的数据进行否定排他逻辑和运算后，将其结果存储到 D200 以后的程序。

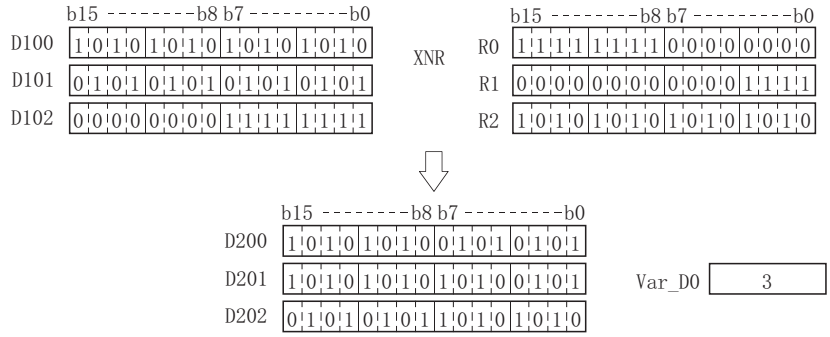
[结构体梯形图]



[ST]

```
BKXNRP(X20,D100,R0,Var_D0,D200);
```

[动作]



7.2 旋转指令

7.2.1 16 位数据的右旋转

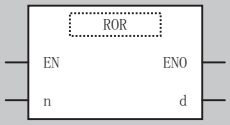
ROR, RCR

Basic | High performance | Universal | L CPU

ROR(P)
RCR(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= ROR (EN, n, d);

中放入下述指令。

ROR	RORP
RCR	RCRP

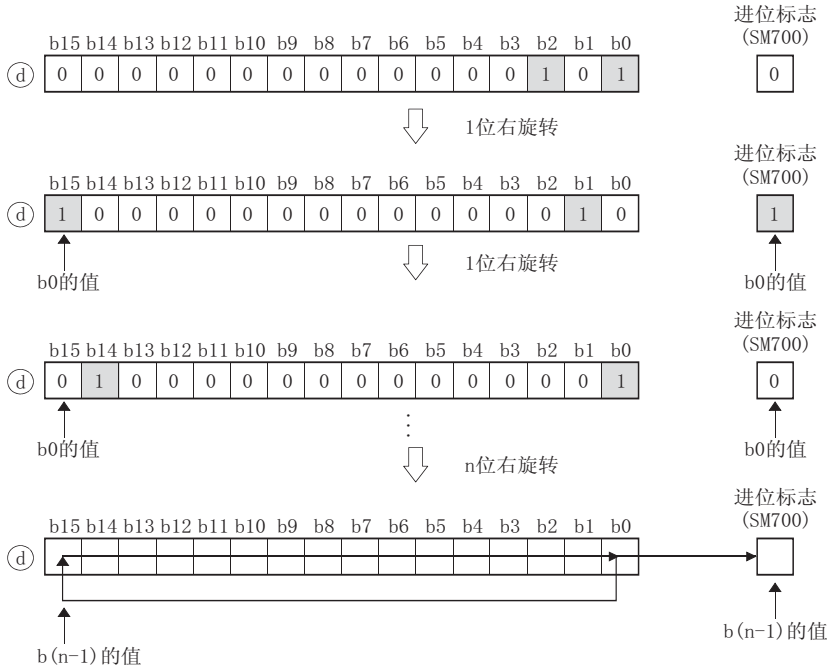
输入自变量, EN: 执行条件 : 位
n: 旋转的次数 (0 ～ 15) : ANY16
输出自变量, ENO: 执行结果 : 位
d: 旋转的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
n		○						○	-
ⓓ		○						-	-

功 能

ROR(P)

(1) 将ⓓ中指定的软元件的 16 位数据在不包含进位标志的状况下向右旋转 n 位。
进位标志的 ON/OFF 取决于 ROR 指令执行之前的状态。

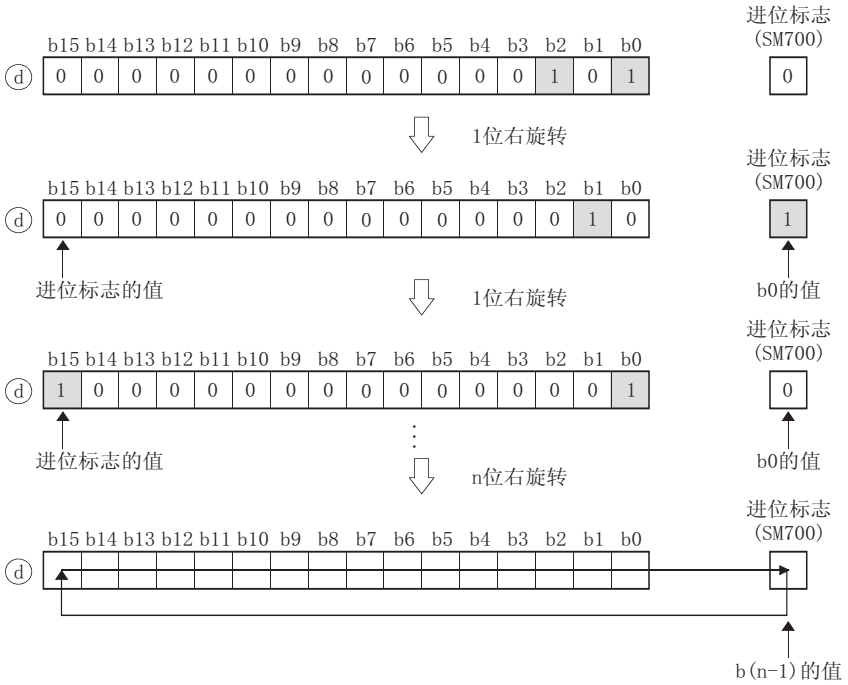


(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为，n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，由于 15 ÷ 12=1 的余数为 3，因此旋转 3 位。

(3) n 中可指定的范围为 0 ～ 15。
n 中指定了 16 以上的值的情况下，将以 n ÷ 16 的余数的值进行旋转。
例如 n=18 时， 由于 18 ÷ 16=1 的余数为 2，因此向右旋转 2 位。

RCR(P)

(1) 将ⓓ中指定的软元件的 16 位数据在包含进位标志的状况下向右旋转 n 位。
进位标志 ON/OFF 取决于 RCR 指令执行之前的状态。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为，n ÷（位数指定中指定的点数）的余数。
例如，n=15，（位数指定中指定的点数）=12 位时，由于 15 ÷ 12=1 的余数为 3，因此旋转 3 位。

(3) n 中可指定的范围为 0 ～ 15。
n 中指定了 16 以上的值的情况下，将以 n ÷ 16 的余数的值进行旋转。
例如 n=18 时， 由于 18 ÷ 16=1 的余数为 2，因此向右旋转 2 位。

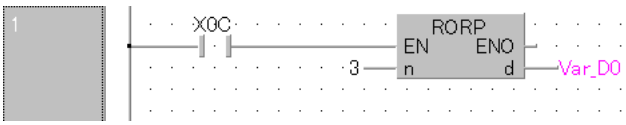
出 错

不存在 ROR(P)、RCR(P) 指令相关的运算出错。

程序示例

(1) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在不包含进位标志的状况下向右旋转 3 位的程序。

[结构体梯形图]



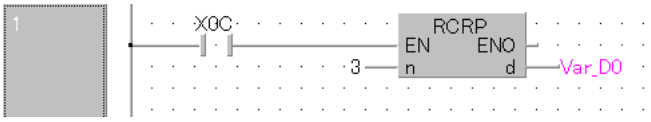
[ST]

```
RORP(X0C,3,Var_D0);
```

[动作]

(2) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在包含进位标志的状况下向右旋转 3 位的程序。

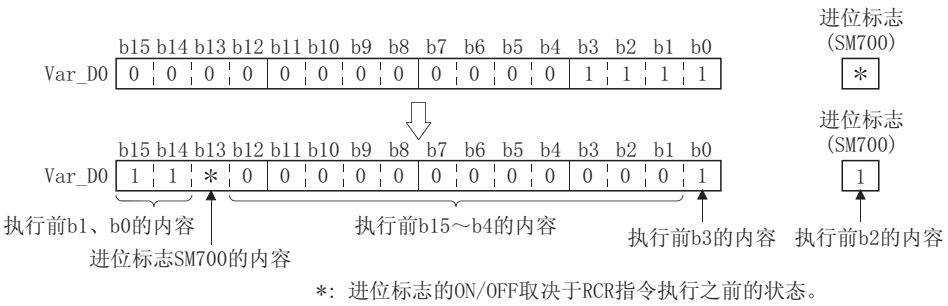
[结构体梯形图]



[ST]

```
RCRP(X0C,3,Var_D0);
```

[动作]



*: 进位标志的ON/OFF取决于RCR指令执行之前的状态。

7.2.2 16 位数据的左旋转

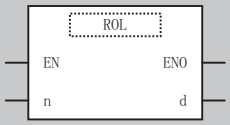
ROL, RCL

Basic | High performance | Universal | L CPU

ROL(P)
RCL(P)

P: 执行条件 : ⏋

结构体梯形图



ST

ENO:= ROL (EN, n, d);

[ ] 中放入下述指令。

ROL	ROLP
RCL	RCLP

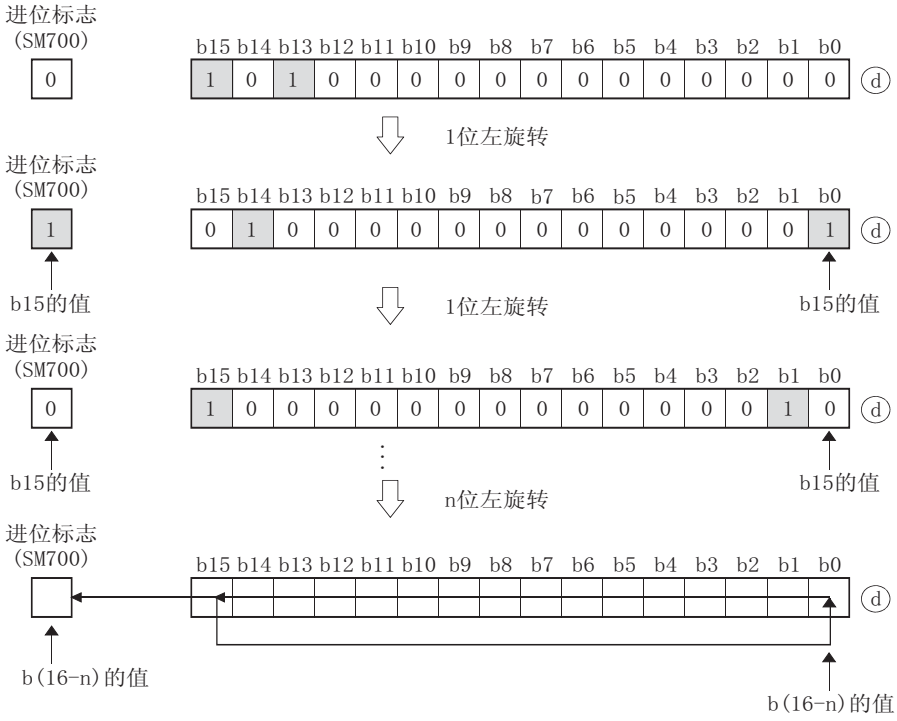
输入自变量, EN: 执行条件 : 位
n: 旋转的次数 (0 ～ 15) : ANY16
输出自变量, ENO: 执行结果 : 位
d: 旋转的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
n				○				○	-
ⓓ				○				-	-

功 能

ROL(P)

(1) 将ⓓ中指定的软元件的 16 位数据在不包含进位标志的状况下向左旋转 n 位。
进位标志的 ON/OFF 取决于 ROL 指令执行之前的状态。



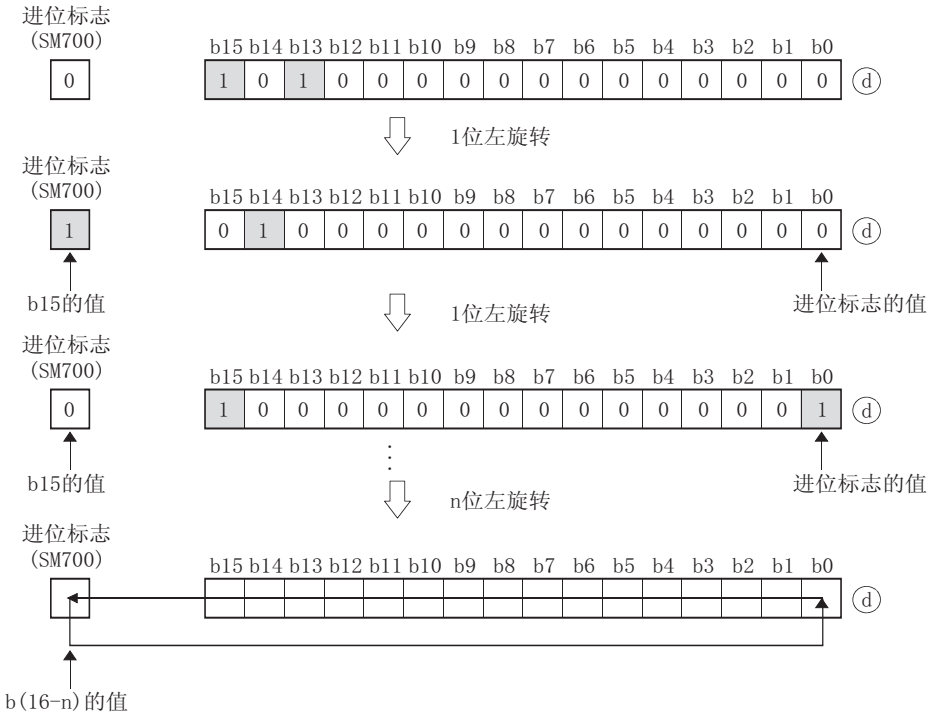
(2) ⓓ中指定了位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为，n ÷（位数指定中指定的点数）的余数。
例如 n=15，（位数指定中指定的点数）=12 位时，由于 15 ÷ 12=1 的余数为 3，因此变为旋转 3 位。

(3) 在 n 中可指定的范围为 0 ～ 15。
在 n 中指定了 16 以上的值的情况下，将以 n ÷ 16 的余数的值进行旋转。
例如 n=18 时， 18 ÷ 16=1 的余数为 2 因此向左旋转 2 位。

RCL(P)

(1) 将ⓓ中指定的软元件的 16 位数据在包含进位标志的状况下向左旋转 n 位。

进位标志的 ON/OFF 取决于 RCL 指令执行之前的状态。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。

此时实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。

例如 n=15，（位数指定中指定的点数）=12 位时，由于 15 ÷ 12=1 的余数为 3，因此旋转 3 位。

(3) n 中可指定的范围为 0 ～ 15。

n 中指定了 16 以上的值的情况下，将以 n ÷ 16 的余数的值进行旋转。

例如 n=18 时， 由于 18 ÷ 16=1 的余数为 2，因此向左旋转 2 位。

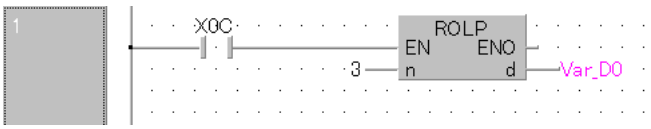
出 错

不存在 ROL(P)、RCL(P) 指令相关的的出错。

程序示例

(1) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在不包含进位标志的状况下向左旋转 3 位的程序。

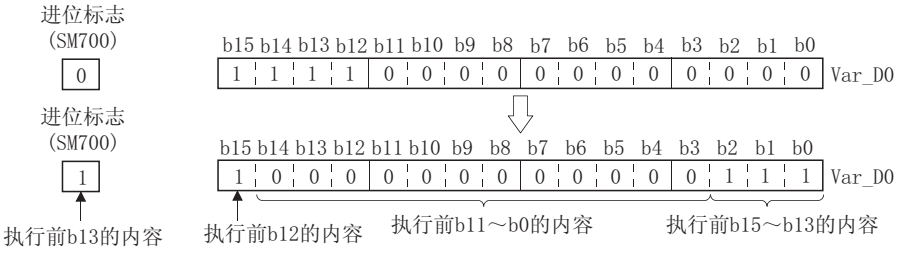
[结构体梯形图]



[ST]

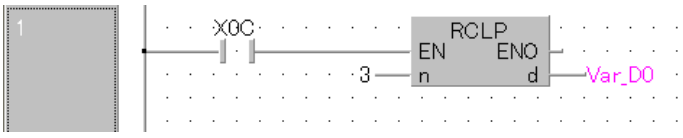
```
ROLP(X0C,3,Var_D0);
```

[动作]



(2) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在包含进位标志的状况下向左旋转 3 位的程序。

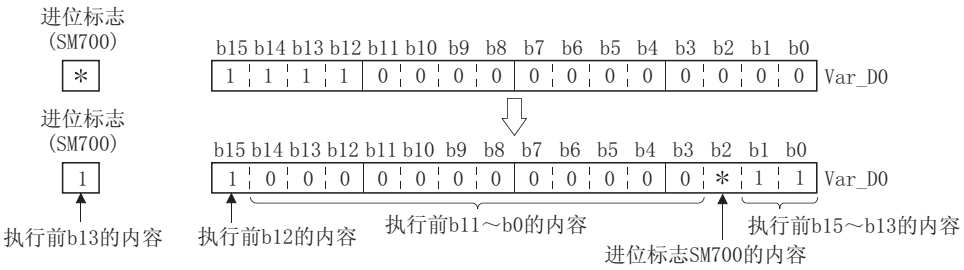
[结构体梯形图]



[ST]

```
RCLP(X0C,3,Var_D0);
```

[动作]



* 进位标志的ON/OFF取决于RCL指令执行之前的状态。

7.2.3 32 位数据的右旋转

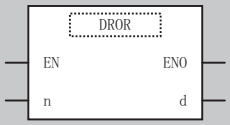
DROR, DRCR

Basic | High performance | Universal | L CPU

DROR(P)
DRCR(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= DROR (EN, n, d);

[ ] 中放入下述指令。

DROR	DRORP
DRCR	DRCRP

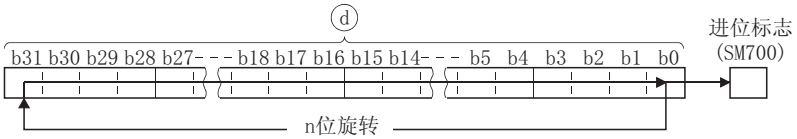
输入自变量， EN: 执行条件 : 位
n: 旋转的次数 (0 ～ 31) : ANY16
输出自变量， ENO: 执行结果 : 位
d: 旋转的软元件的起始编号 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
n		○						○	-
ⓓ		○						-	-

功 能

DROR(P)

(1) 将ⓓ中指定的软元件的 32 位数据在不包含进位标志的状况下向右旋转 n 位。
进位标志的 ON/OFF 取决于 DROR 指令执行之前的状态。

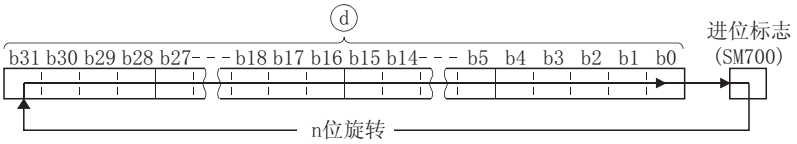


(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如 n=31，（位数指定中指定的点数）=24 位时，由于 31 ÷ 24=1 的余数为 7，因此旋转 7 位。

(3) n 中可指定的范围为 0 ～ 31。
n 中指定了 32 以上的值的情况下，将以 n ÷ 32 的余数的值进行旋转。
例如 n=34 时，由于 34 ÷ 32=1 的余数为 2，因此向右旋转 2 位。

DRCR(P)

(1) 将ⓓ中指定的软元件的 32 位数据在包含进位标志的状况下向右旋转 n 位。
进位标志的 ON/OFF 取决于 DRCR 指令执行之前的状态。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如 n=31，（位数指定中指定的点数）=24 位时，由于 31 ÷ 24=1 的余数为 7，因此旋转 7 位。

(3) n 中可指定的范围为 0 ～ 31。
n 中指定了 32 以上的值的情况下，将以 n ÷ 32 的余数的值进行旋转。
例如 n=34 时，由于 34 ÷ 32=1 的余数为 2，因此向右旋转 2 位。

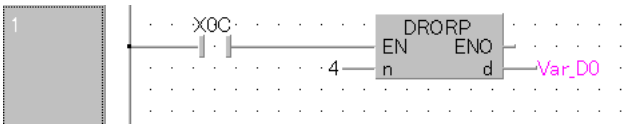
出 错

不存在 DROR(P)、DRCR(P) 指令相关的出错。

程序示例

(1) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在不包含进位标志的状况下向右旋转 4 位的程序。

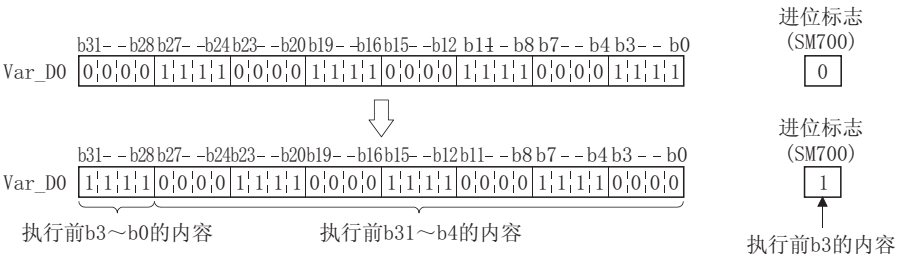
[结构体梯形图]



[ST]

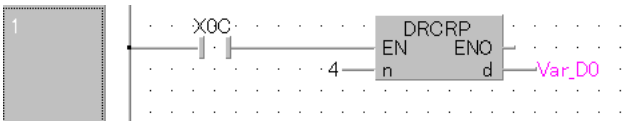
```
DRORP(X0C,4,Var_D0);
```

[动作]



(2) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在包含进位标志的状况下向右旋转 4 位的程序。

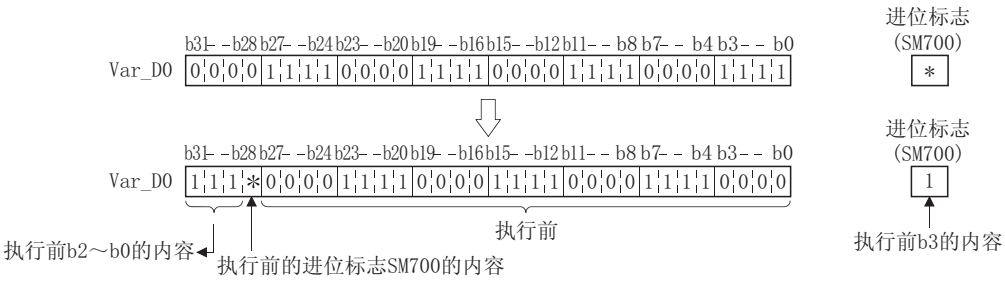
[结构体梯形图]



[ST]

```
DRCRP(X0C,4,Var_D0);
```

[动作]



*: 进位标志的ON/OFF取决于DRCR指令执行之前的状态。

7.2.4 32 位数据的左旋转

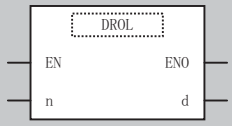
DROL, DRCL

Basic | High performance | Universal | L CPU

DROL(P)
DRCL(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= DROL (EN, n, d);
```

[ ] 中放入下述指令。

DROL	DROLP
DRCL	DRCLP

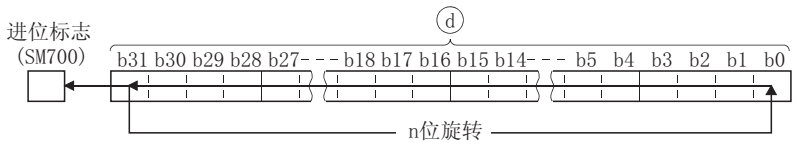
输入自变量， EN: 执行条件 : 位
n: 旋转的次数 (0 ～ 31) : ANY16
输出自变量， ENO: 执行结果 : 位
d: 旋转的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n		○						○	-
ⓓ		○						-	-

功 能

DROL(P)

(1) 将ⓓ中指定的软元件的 32 位数据在不包含进位标志的状况下向左旋转 n 位。
进位标志的 ON/OFF 取决于 DROL 指令执行之前的状态。

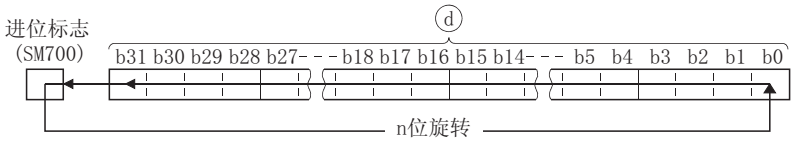


(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如 n=31，（位数指定中指定的点数）=24 位时，由于 31 ÷ 24=1 的余数为 7，因此旋转 7 位。

(3) n 中可指定的范围为 0 ～ 31。
n 中指定了 32 以上的值的情况下，以 n ÷ 32 的余数的值进行旋转。
例如 n=34 时，由于 34 ÷ 32=1 的余数为 2，因此向左旋转 2 位。

DRCL(P)

(1) 将ⓓ中指定的软元件的 32 位数据在包含进位标志的状况下向左旋转 n 位。
进位标志的 ON/OFF 取决于 DRCL 指令执行之前的状态。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围进行旋转。
此时实际旋转的位数为 n ÷（位数指定中指定的点数）的余数。
例如 n=31，（ 位数指定中指定的点数）=24 位时，由于 31 ÷ 24=1 的余数为 7，因此旋转 7 位。

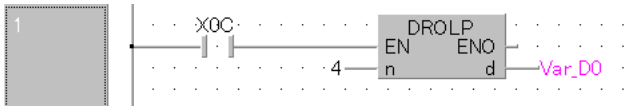
(3) n 中可指定的范围为 0 ～ 31。
n 中指定了 32 以上的值的情况下，以 n ÷ 32 的余数的值进行旋转。
例如 n=34 时，由于 34 ÷ 32=1 的余数为 2，因此向左旋转 2 位。

出 错

不存在 DROL(P)、DRCL(P) 指令相关的出错。

程序示例

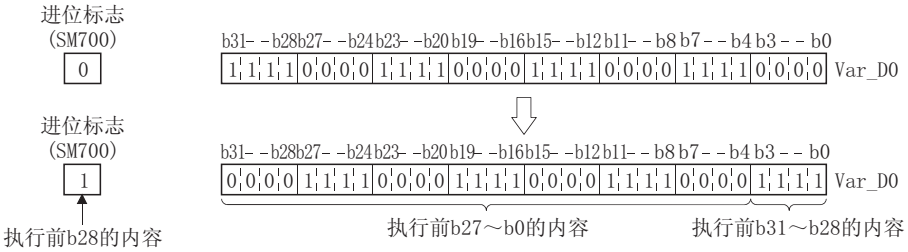
(1) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在不包含进位标志的状况下向左旋转 4 位的程序。
[结构体梯形图]



[ST]

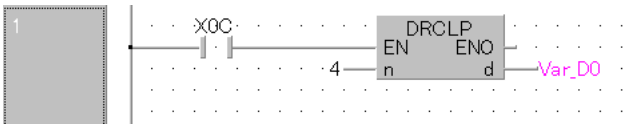
```
DROLP(X0C,4,Var_D0);
```

[动作]



(2) 以下为 X0C 变为 ON 时，将 Var_D0 的内容在包含进位标志的状况下向左旋转 4 位的程序。

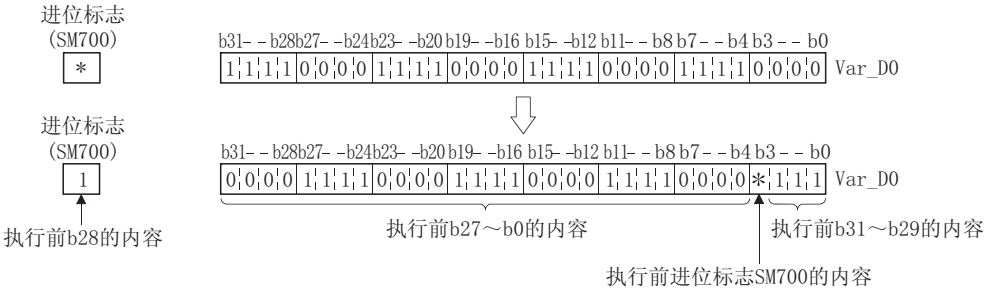
[结构体梯形图]



[ST]

```
DRCLP(X0C,4,Var_D0);
```

[动作]



*: 进位标志的ON/OFF取决于DRCL指令执行之前的状态。

7.3 移动指令

7.3.1 16 位数据的 n 位右移动、左移动

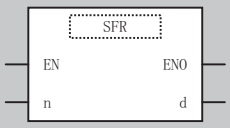
SFR, SFL

Basic | High performance | Universal | L CPU

SFR(P)
SFL(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= SFR (EN, n, d);

中放入下述指令。

SFR SFRP
SFL SFLP

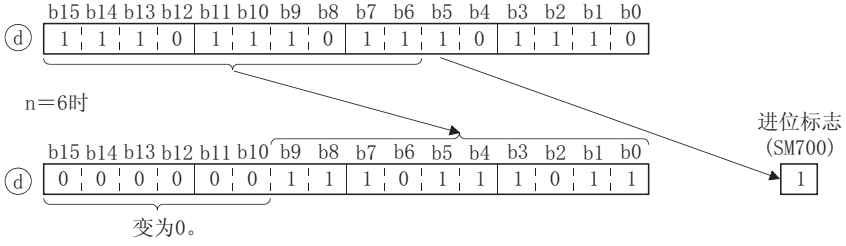
输入自变量， EN: 执行条件 : 位
n: 移动的次数 (0 ～ 15) : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储移动数据的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
n		○						○	-
ⓓ		○						-	-

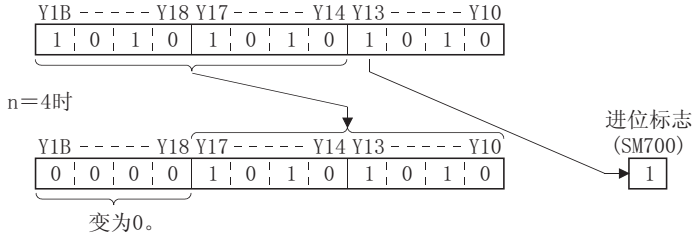
功 能

SFR(P)

(1) 将ⓓ中指定的软元件的 16 位数据向右移动 n 位。
最高位算起的 n 位变为 0。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围向右移动。

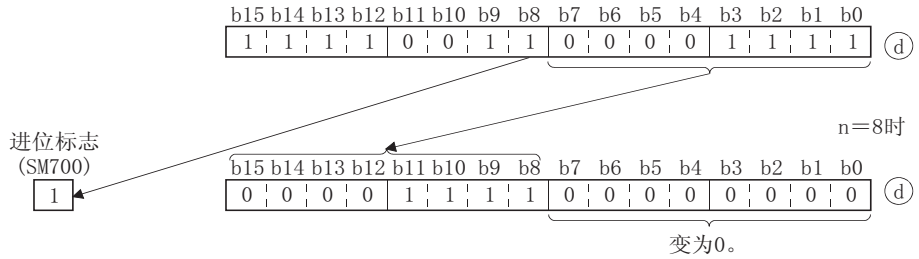


此时实际移动的位数为，n ÷（位数指定中指定的点数）的余数。
例如 n=15，（位数指定中指定的点数）=8 位时，由于 15 ÷ 8=1 的余数为 7，以移动 7 位。

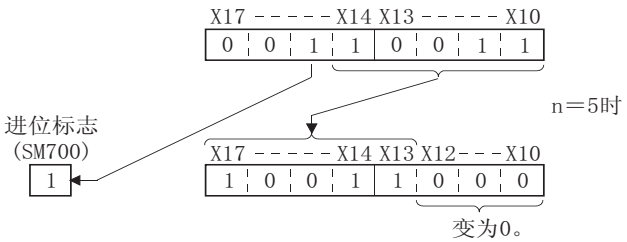
(3) n 中可指定的范围为 0 ～ 15。
n 中指定了 16 以上的值的情况下，以 n ÷ 16 的余数的值向右移动。
例如 n=18 时， 由于 18 ÷ 16=1 的余数为 2，因此向右移动 2 位。

SFL(P)

(1) 将ⓓ中指定的软元件的 16 位数据向左移动 n 位。
最低位算起的 n 位变为 0。



(2) ⓓ中指定位软元件的情况下，以位数指定中指定的软元件范围向左移动。



此时实际移动的位数为，n ÷（位数指定中指定的点数）的余数。
例如 n=15，（位数指定中指定的点数）=8 位时，由于 15 ÷ 8=1 的余数为 7，因此移动 7 位。

(3) n 中可指定的范围为 0 ～ 15。
n 中指定了 16 以上的值的情况下，以 n ÷ 16 的余数的值向左移动。
例如 n=18 时， 由于 18 ÷ 16=1 的余数为 2，因此向左移动 2 位。

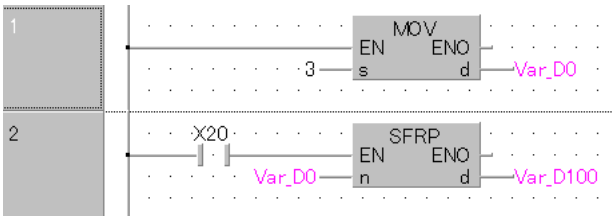
出 错

不存在 SFR(P)、SFL(P) 指令相关的运算出错。

程序示例

(1) 以下为 X20 变为 ON 时，将 Var_D0 的内容以 Var_D100 中指定的位向右移动的程序。

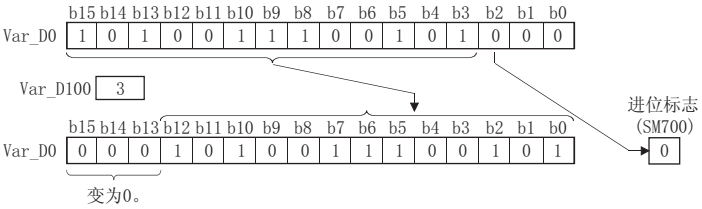
[结构体梯形图]



[ST]

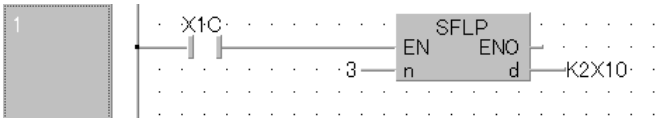
```
Var_D0:=3;
SFRP(X20,Var_D0,Var_D100);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将 X10 ～ X17 的内容向左移动 3 位的程序。

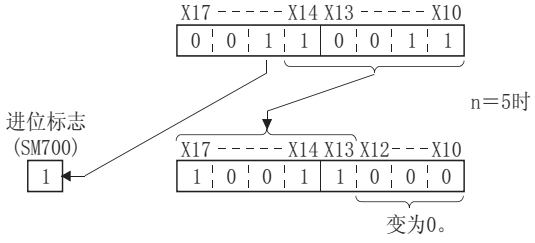
[结构体梯形图]



[ST]

```
SFLP(X1C,3,K2X10);
```

[动作]



7.3.2 n位数据的1位右移动、左移动

BSFR, BSFL

Basic | High performance | Universal | L CPU

BSFR(P)
BSFL(P)

P: 执行条件 : ↑

结构体梯形图

ST

ENO:= BSFR (EN, n, d);

中放入下述指令。

BSFR	BSFRP
BSFL	BSFLP

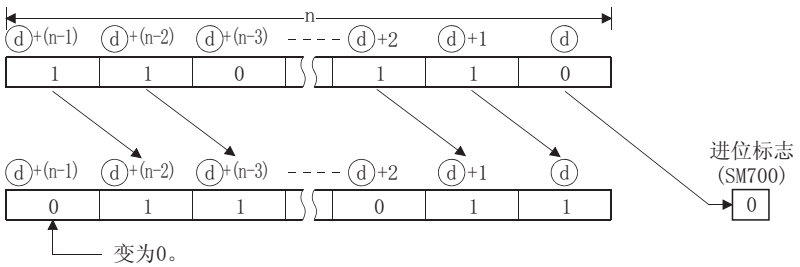
输入自变量， EN: 执行条件 : 位
n: 移动的软元件的数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 移动的软元件的起始编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n	○			○					-
ⓓ	○			-					-

功 能

BSFR(P)

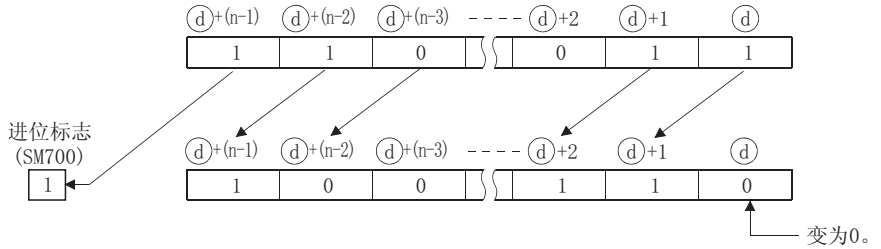
(1) 将ⓓ中指定的软元件算起的n点向右移动1位。



(2) 在ⓓ+(n-1)中指定的软元件将变为0。

BSFL(P)

(1) 将ⓓ中指定的软元件算起的 n 点向左移动 1 位。



(2) ⓓ中指定的软元件将变为 0。

出 错

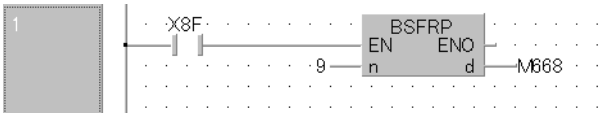
在以下情况下，将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ的软元件算起 n 点的范围超出了相应软元件时。 （出错代码： 4101）

程序示例

(1) 以下为 X8F 变为 ON 时，将 M668 ～ M676 的数据向右移动的程序。

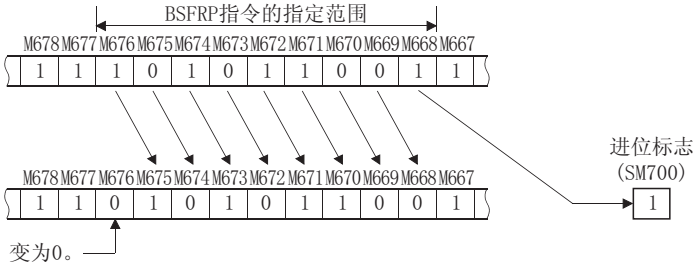
[结构体梯形图]



[ST]

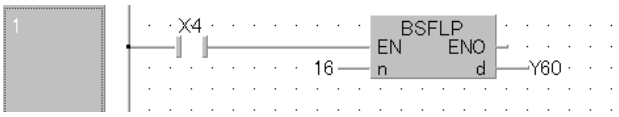
```
BSFRP(X8F,9,M668);
```

[动作]



(2) 以下为 X4 变为 ON 时，将 Y60 ～ Y6F 的数据向左移动的程序。

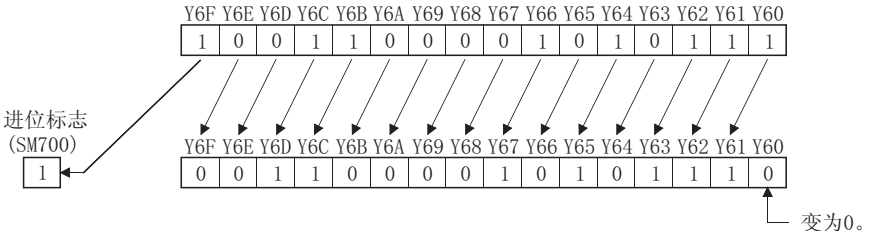
[结构体梯形图]



[ST]

```
BSFLP(X4,16,Y60);
```

[动作]



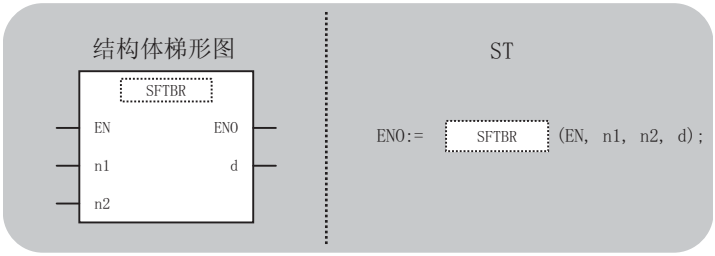
7.3.3 n位数据的n位右移动、左移动

SFTBR, SFTBL

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前5位数为“10102”以后
- QnUDE(H)CPU：序列号的前5位数为“10102”以后

SFTBR(P)
SFTBL(P)

（P：执行条件　　：）



[　　] 中放入下述指令。

SFTBR	SFTBRP
SFTBL	SFTBLP

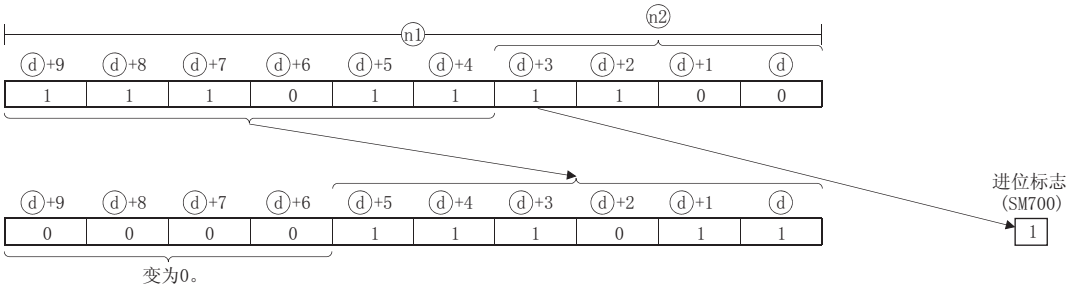
输入自变量，EN：　执行条件　：位
　　n1：　移动的软元件的数　：ANY16
　　n2：　移动数　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　移动的软元件的起始编号　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n1	－	○	○			○			－
n2	－	○	○			○			－
ⓓ	○	－	○			－			－

功 能

SFTBR(P)

(1) 将ⓓ中指定的软元件算起n1位的数据向右移动n2位。
n1=10，n2=4时



(2) 对于 n1、n2 应指定为 n1>n2。n1 ≦ n2 的情况下，将以 n2 ÷ n1 的余数的值进行移动。
但是，n2 ÷ n1 的余数的值为 0 时，变为无处理。

(3) n1 的可设置范围为 1 ～ 64。

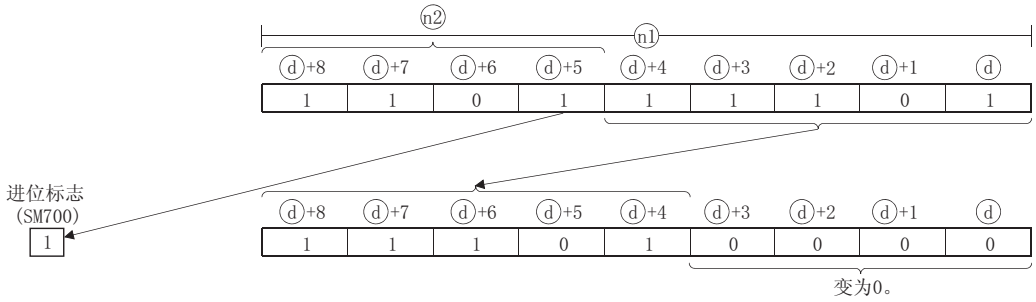
(4) 最高位算起的 n2 位将变为 0。n1 ＜ n2 的情况下，n2 ÷ n1 的余数的值的部分将变为 0。

(5) n1 或 n2 中指定的值为 0 时，将变为无处理。

SFTBL(P)

(1) 将ⓓ中指定的软元件算起的 n1 位的数据向左移动 n2 位。

n1=10，n2=4 时



(2) 对于 n1、n2 应指定为 n1>n2。n1<n2 的情况下，将以 n2 ÷ n1 的余数的值进行移动。
但是，n2 ÷ n1 的余数的值为 0 时，将变为无处理。

(3) n1 中可设置的范围为 1 ～ 64。

(4) 最低位算起的 n2 位将变为 0。n1<n2 的情况下，n2 ÷ n1 的余数的值的部分将变为 0。

(5) n1 或 n2 中指定的值为 0 时，将变为无处理。

出 错

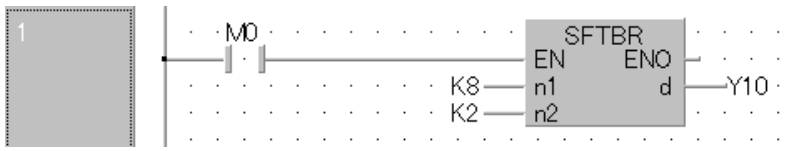
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n1 超出了 0 ～ 64 的范围时。 (出错代码： 4100)
- n2 为负数时。 (出错代码： 4100)
- n1 中指定的软元件点数超出了ⓓ中指定的软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将④中指定的 Y10 至 Y17(8 位) 的内容向右移动 2(n2) 位的程序。

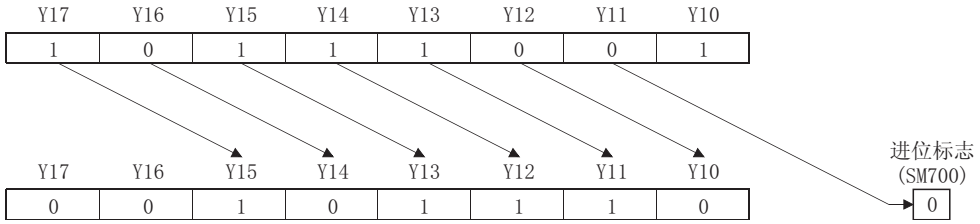
[结构体梯形图]



[ST]

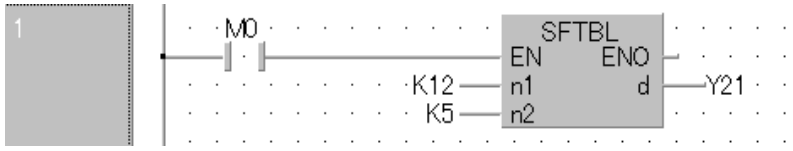
```
SFTBR(M0,K8,K2,Y10);
```

[动作]



(2) 以下为 M0 变为 ON 时，将④中指定的 Y21 至 Y2C(12 位) 的内容向左移动 5(n2) 位的程序。

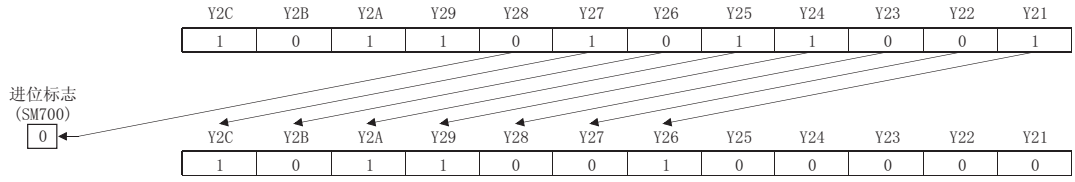
[结构体梯形图]



[ST]

```
SFTBL(M0,K12,K5,Y21);
```

[动作]



7.3.4 n字数据的1字右移动、左移动

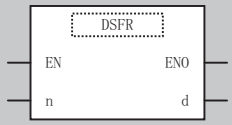
DSFR, DSFL

Basic | High performance | Universal | L CPU

DSFR(P)
DSFL(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= DSFR (EN, n, d);

中放入下述指令。

DSFR	DSFRP
DSFL	DSFLP

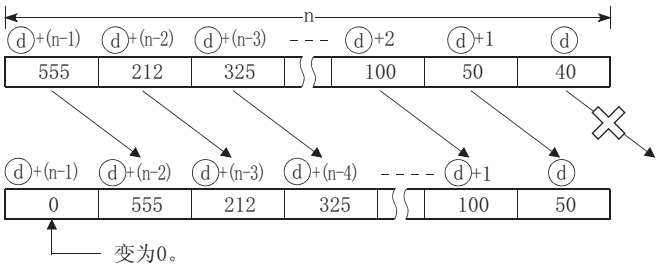
输入自变量， EN: 执行条件 : 位
n: 移动的软元件的数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 移动的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n	○	○		○					-
ⓓ	-	○		-					-

功 能

DSFR(P)

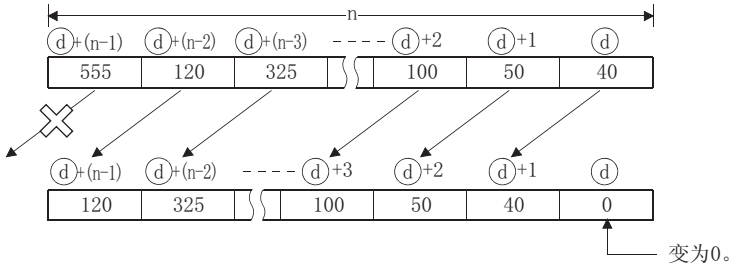
(1) 将ⓓ中指定的软元件算起的n点向右移动1字。



(2) ⓓ+(n-1)中指定的软元件将变为0。

DSFL(P)

(1) 将ⓓ中指定的软元件算起的 n 点向左移动 1 字。



(2) ⓓ中指定的软元件将变为 0。

出 错

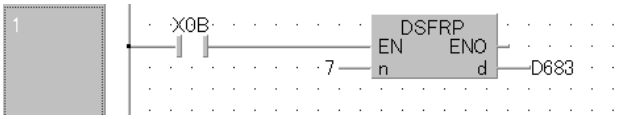
在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ的软元件算起 n 点的范围超出了相应软元件时。 (出错代码 : 4101)

程序示例

(1) 以下为 X0B 变为 ON 时，将 D683 ～ D689 的内容向右移动的程序。

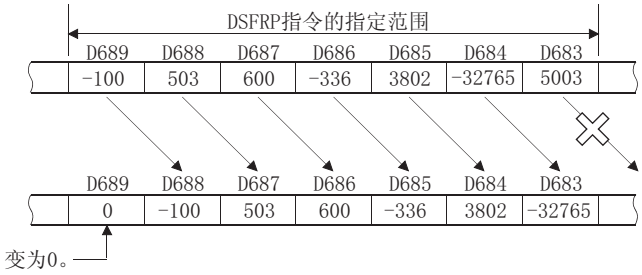
[结构体梯形图]



[ST]

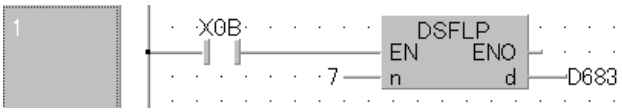
```
DSFRP(X0B,7,D683);
```

[动作]



(2) 以下为 X0B 变为 ON 时，将 D683 ～ D689 的内容向左移动的程序。

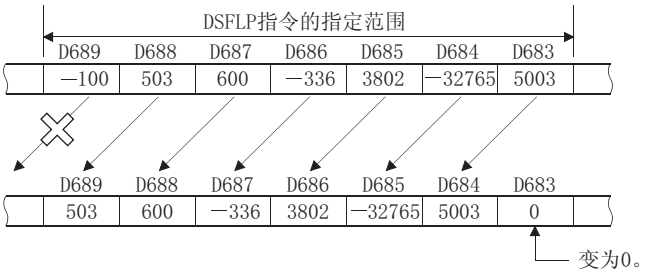
[结构体梯形图]



[ST]

```
DSFLP(X0B,7,D683);
```

[动作]



7.3.5 n字数据的n字右移动、左移动

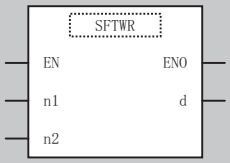
SFTWR, SFTWL

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前5位数为“10102”以后
- QnUDE(H)CPU：序列号的前5位数为“10102”以后

SFTWR(P)
SFTWL(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:=  SFTWR  (EN, n1, n2, d);
```

中放入下述指令。

SFTWR	SFTWRP
SFTWL	SFTWLP

输入自变量，	EN:	执行条件	：位
	n1:	移动的字数	：ANY16
	n2	移动数	：ANY16
输出自变量，	ENO:	执行结果	：位
	d:	移动的软元件的起始编号	：ANY16

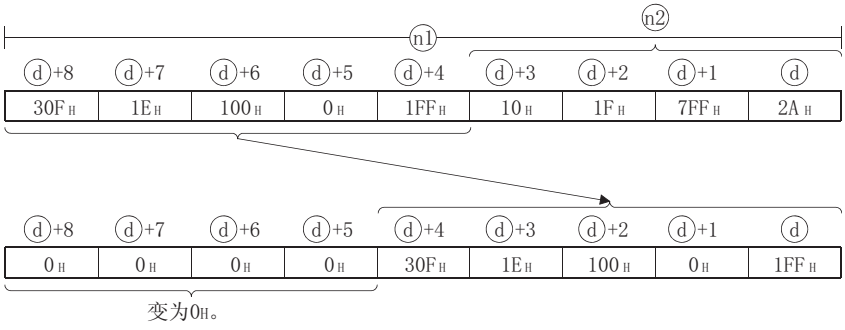
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n1	－	○	○			○			－
n2	－	○	○			○			－
ⓓ	－	○	○			－			－

功 能

SFTWR(P)

(1) 将ⓓ中指定的软元件算起的n1字的数据向右移动n2字。

n1=9，n2=4时



(2) 最高位算起的 n2 字将变为 0。

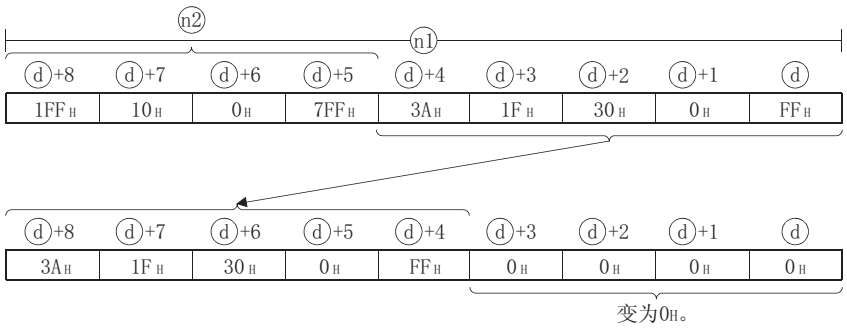
(3) n1 或 n2 中指定的值为 0 时，将变为无处理。

(4) n1 ≦ n2 的情况下，ⓓ中指定的软元件算起的 n1 字的数据将全部变为 0。

SFTWL(P)

(1) 将ⓓ中指定的软元件算起的 n1 字的数据向左移动 n2 字。

n1=9，n2=4 时



(2) 最低位算起的 n2 字将变为 0。

(3) n1 或 n2 中指定的值为 0 时，将变为无处理。

(4) n1 ≦ n2 的情况下，ⓓ中指定的软元件算起的 n1 字的数据将全部变为 0。

出 错

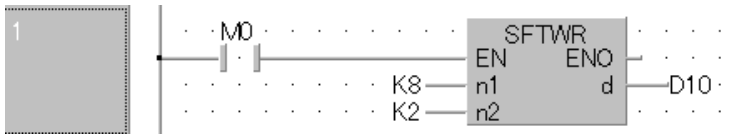
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n1 或 n2 为负数时。 (出错代码： 4100)
- n1 中指定的软元件点数超出了ⓓ中指定的软元件的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，将ⓓ中指定的 D10 至 8(n1) 字的内容向右移动 2(n2) 字的程序。

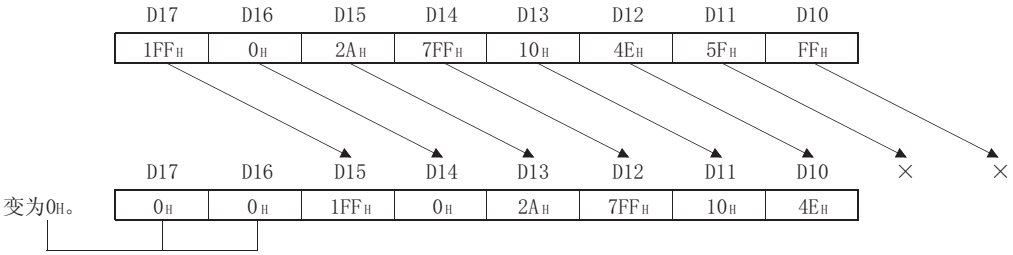
[结构体梯形图]



[ST]

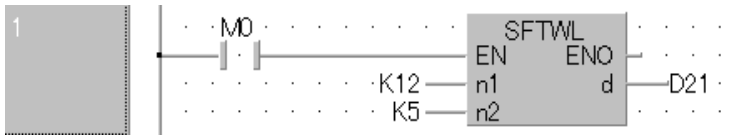
```
SFTWR(M0,K8,K2,D10);
```

[动作]



(2) 以下为 M0 变为 ON 时，将ⓓ中指定的 D21 至 12(n1) 字的内容向左移动 5(n2) 字的程序

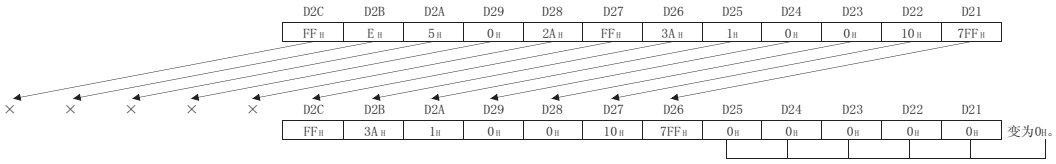
[结构体梯形图]



[ST]

```
SFTWL(M0,K12,K5,D21);
```

[动作]



7.4 位处理指令

7.4.1 字软元件的位设置 / 复位

BSET, BRST

Basic | High performance | Universal | L CPU

BSET(P)
BRST(P)

P: 执行条件 : ↑

结构体梯形图

ST

ENO:= BSET (EN, n, d);

□ 中放入下述指令。

BSET	BSETP
BRST	BRSTP

输入自变量， EN: 执行条件 : 位
n: 进行位设置、复位的位编号 (0 ～ 15) : ANY16
输出自变量， ENO: 执行结果 : 位
d: 进行位设置、复位的软元件编号 : ANY16

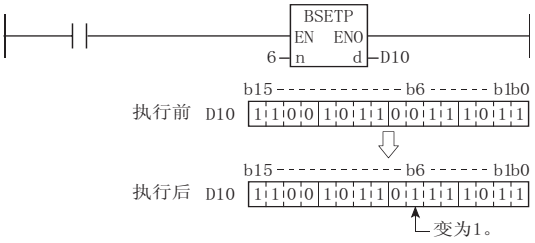
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n				○				○	-
ⓓ				○				-	-

功 能

BSET(P)

(1) 对ⓓ中指定的字软元件的第 n 位进行设置 (1)。

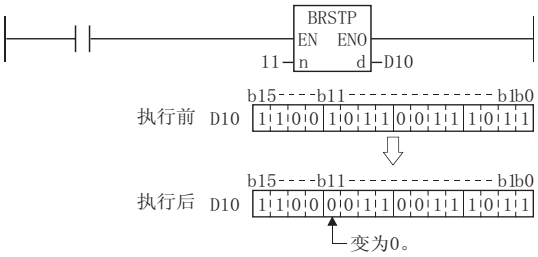
(2) n 中超过了 15 的情况下，将以低 4 位的数据执行。



BRST(P)

(1) 对ⓓ中指定的字软元件的第 n 位进行复位 (0)。

(2) n 中超过了 15 的情况下，以低 4 位的数据执行。



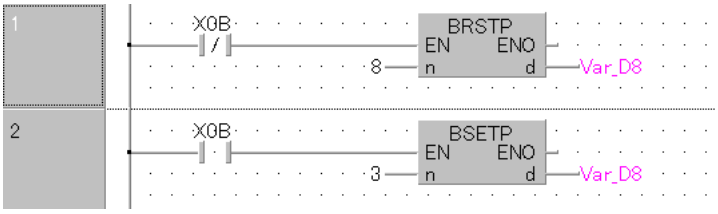
出 错

不存在 BSET(P)、BRST(P) 指令相关的运算出错。

程序示例

以下为 X0B 为 OFF 时，对 Var_D8 的第 8 位 (b8) 进行复位 (0)，X0B 变为 ON 时，对 D8 的第 3 位 (b3) 进行设置 (1) 的程序。

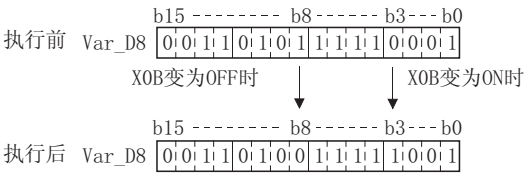
[结构体梯形图]



[ST]

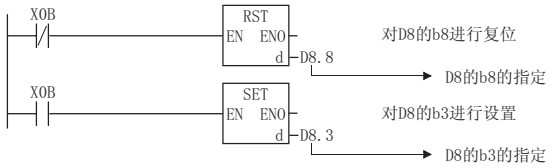
```
BRSTP(NOT(X0B),8,Var_D8);
BSETP(X0B,3,Var_D8);
```

[动作]



备 注

对于字软元件的位设置 / 复位，也可以通过字软元件的位指定进行。
关于字软元件的位指定，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
对于程序示例 (1) 的处理，如果使用字软元件的位指定，则其情况如下所示。



7.4.2 位测试

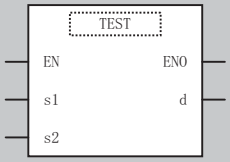
TEST, DTEST

Basic | High performance | Universal | L CPU

TEST(P)
DTEST(P)

P: 执行条件 : ↵

结构体梯形图



ST

ENO:= TEST (EN, s1, s2, d);

[ ] 中放入下述指令。

TEST	TESTP
DTEST	DTESTP

输入自变量, EN: 执行条件 : 位
s1: 要截取的位数据 : ANY16
s2: 要截取的位数据的位置 : ANY16
(0 ～ 15(TEST)/0 ～ 31(DTEST))
输出自变量, ENO: 执行结果 : 位
d: 存储截取的位数据的位软元件编号 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	○						○	-	-
ⓢ2	○						○	○	-
ⓓ	○						-	-	-

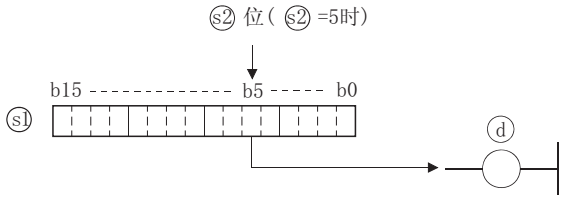
功 能

TEST(P)

(1) 将ⓢ1 中指定的字软元件内的ⓢ2 中指定的位置的位数据进行截取后，写入到ⓓ 中指定的位软元件中。

(2) 对于ⓓ 中指定的位软元件，相应位为 0 时置为 OFF，为 1 时置为 ON。

(3) 在ⓢ2 中指定的位置处，对 1 字数据的各个位位置 (0 ～ 15) 进行指定。
ⓢ2 中指定为 16 以上的情况下，n ÷ 16 的余数值的位置的位数据将成为对象。
例如 n=18 时，由于 18 ÷ 16=1 的余数为 2，因此变为 b2 的数据。

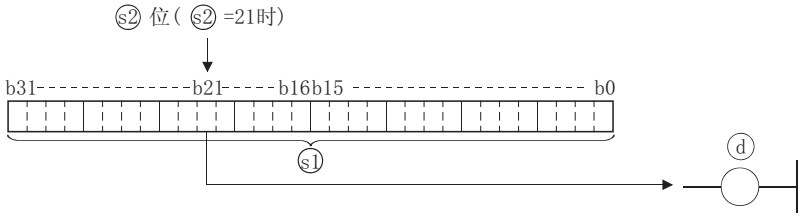


DTEST(P)

(1) 将ⓢ1 中指定的 2 字软元件内的ⓢ2 中指定的位置的位数据进行截取后，写入到ⓓ 中指定的位软元件中。

(2) 对于ⓓ 中指定的位软元件，相应位为 0 时置为 OFF，为 1 时置为 ON。

(3) 在ⓢ2 中指定的位置处，对 2 字数据的各个位位置 (0 ～ 31) 进行指定。
ⓢ2 中指定为 32 以上的情况下，n ÷ 32 的余数值的位置的位数据将成为对象。
例如 n=34 时，由于 34 ÷ 32=1 的余数为 2，因此变为 b2 的数据。



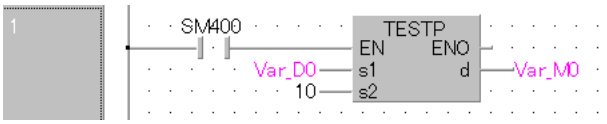
出 错

不存在 TEST(P)、DTEST(P) 指令相关的运算出错。

程序示例

(1) 以下为根据 1 字数据 (Var_D0) 的第 10 位的状态，将 Var_M0 置为 ON/OFF 的程序。

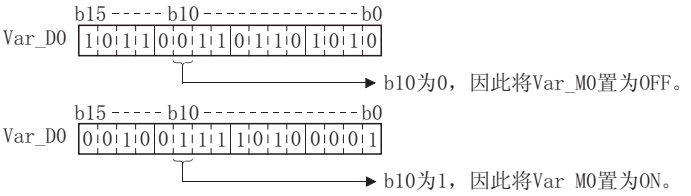
[结构体梯形图]



[ST]

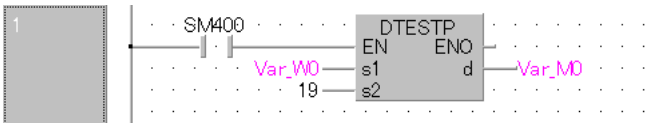
```
TESTP(SM400,Var_D0,10,Var_M0);
```

[动作]



(2) 以下为根据 2 字数据 (Var_W0) 的第 19 位的状态，将 Var_M0 置为 ON/OFF 的程序。

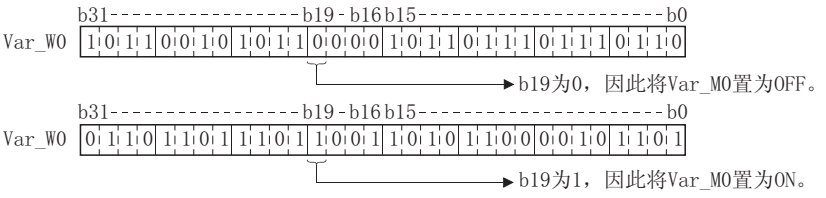
[结构体梯形图]



[ST]

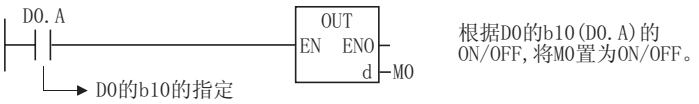
```
DTESTP(SM400,Var_W0,19,Var_M0);
```

[动作]



备 注

对于使用了位测试指令的程序，可以替换为使用了字软元件的位指定的程序。
如果将程序示例 (1) 的程序使用字软元件的位指定，则其情况如下所示。



7.4.3 位软元件的批量复位

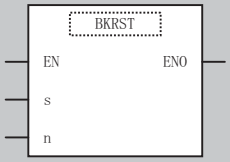
BKRST

Basic | High performance | Universal | L CPU

BKRST(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BKRST (EN, s, n);

□ 中放入下述指令。

BKRST　　BKRSTP

输入自变量，EN：　执行条件　：位
　　s：　复位的软元件的起始编号　：位
　　n：　复位的软元件数　：ANY16
输出自变量，ENO：　执行结果　：位

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 K，H	其它
	位	字		位	字				
ⓢ	○			－					－
n	○			○					－

功 能

(1) 对ⓢ中指定的位软元件算起 n 点的位软元件进行复位。

软元件	状态
报警器 (F)	• 将ⓢ中指定的报警器 (F) 编号算起的 n 点置为 OFF。 • 将变为 OFF 后的报警器编号从 SD64 ～ SD79 中删除后，向前对齐。 • 将 SD64 ～ SD79 中存储的报警器数存储到 SD63 中。
定时器 (T) 计数器 (C)	• 将ⓢ中指定的定时器 (T) 或计数器 (C) 编号算起 n 点的当前值置为 0，将线圈触点置为 OFF。
除上述以外的位软元件	• 将ⓢ中指定的软元件算起 n 点的线圈、触点置为 OFF。

(2) 指定的软元件为 OFF 的情况下，软元件的状态不变化。

出 错

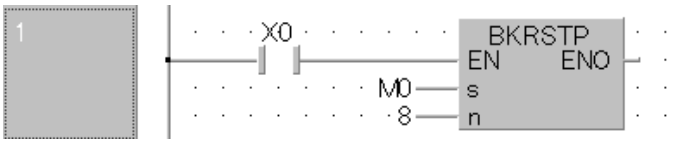
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ的软元件算起 n 点的范围超出了相应软元件时。　　(出错代码：4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 M0 至 M7 置为 OFF 的程序。

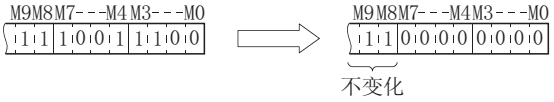
[结构体梯形图]



[ST]

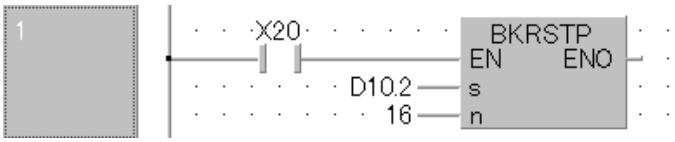
```
BKRSTP(X0,M0,8);
```

[动作]



(2) 以下为 X20 变为 ON 时，将 D10 的第 2 位 (b2) 至 D11 的第 1 位 (b1) 置为 0 的程序。

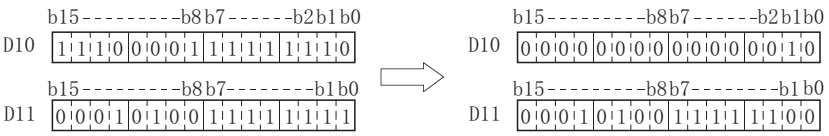
[结构体梯形图]



[ST]

```
BKRSTP(X20,D10.2,16);
```

[动作]



7.5 数据处理指令

7.5.1 16 位 /32 位数据查找

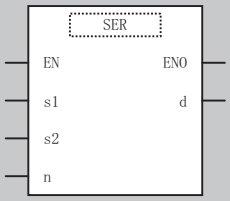
SER, DSER

Basic | High performance | Universal | L CPU

SER(P)
DSER(P)

P: 执行条件 : ⏋

结构体梯形图



ST

ENO:= SER (EN, s1, s2, n, d);

中放入下述指令。

SER	SERP
DSER	DSERP

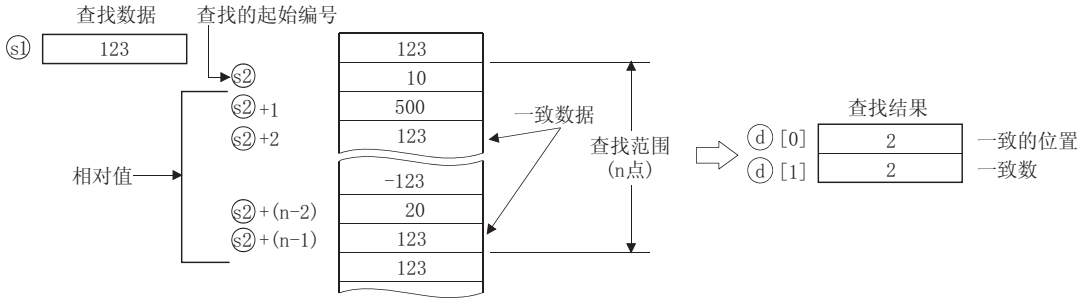
输入自变量， EN: 执行条件 : 位
s1: 查找数据 : ANY16/32
s2: 被查找的数据 : ANY16/32
n: 查找数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储查找结果的软元件的起始编号 : ANY16 的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它
	位	字		位	字				
⑤1	○	○		○	○			○	-
⑤2	-	○		-	-			-	-
n	○	○		○	○			○	-
ⓓ	-	○		-	○			-	-

功 能

SER(P)

(1) 将ⓢ1 中指定的软元件的 16 位数据作为关键字，对ⓢ2 中指定的软元件的 16 位数据开始的 n 点进行查找。
将与关键字一致的个数存储到ⓓ[1] 中指定的软元件中，将最初一致的软元件编号与ⓢ2 的相对值存储到ⓓ中指定的软元件中。

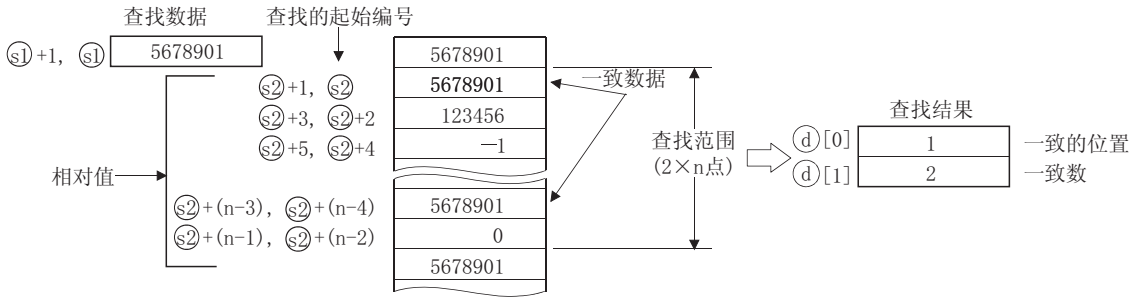


(2) n 为 0 或负数时，将变为无处理。

(3) 查找的结果为未发现一致的数据的情况下，ⓓ中指定的软元件将变为 0。

DSER(P)

(1) 将ⓢ1 中指定的软元件的 32 位数据作为关键字，对ⓢ2 中指定的软元件开始的以 32 位为单位的 n 点（以 16 位为单位的 2 × n 点）进行查找。
将与关键字一致的个数存储到ⓓ[1] 中指定的软元件中，将最初一致的软元件编号与ⓢ2 的相对值存储到ⓓ中指定的软元件中。



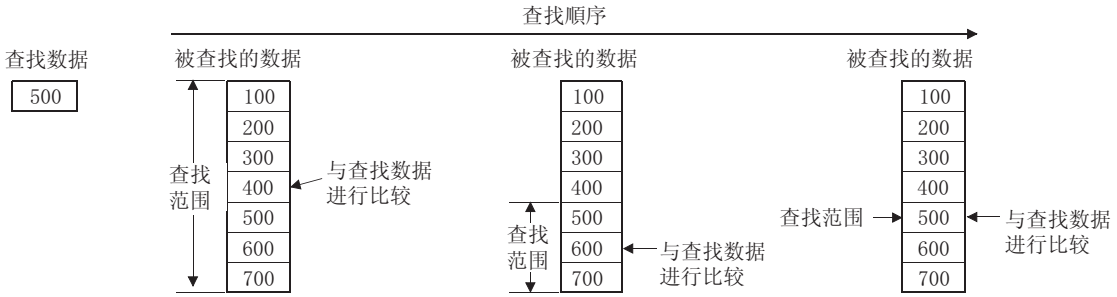
(2) n 为 0 或负数时，将变为无处理。

(3) 查找的结果为未发现一致的数据的情况下，ⓓ中指定的软元件将变为 0。

要 点

在 SER(P)、DSER(P) 指令中，被查找的数据以升序排序的情况下，如果将 SM702[*1] 置为 ON，则可以通过二分查找法加快查找的处理速度。
被查找的数据未以升序排序的情况下，如果将 SM702 置为 ON，则无法得到正常的查找结果。

*1 :SM702 是用于对查找方法进行设置的特殊继电器。
- SM702 OFF: 逐次查找法（线型查找法）
（从被查找的数据的起始开始与查找数据进行比较的方法。）
- SM702 ON : 二分查找法
（对于以升序排序的数据，找出查找范围中间值，根据该值与要查找的值的大小关系确定查找方向以缩小查找范围。反复执行此动作以找出要查找的数据的方法。）



出 错

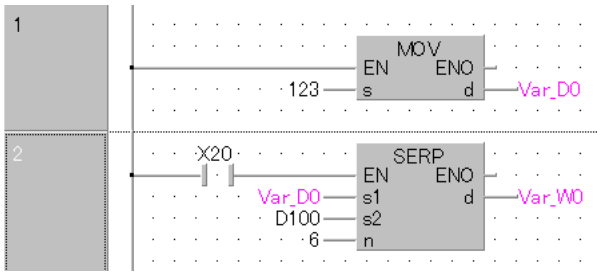
在以下情况下将变为出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ㉒ 的软元件算起的 n 点超出了指定软元件范围时。 （出错代码： 4101）
- ㉔ 中指定的软元件超出了相应软元件的范围时。
（通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

(1) 以下为 X20 变为 ON 时，对 D100 ～ D105 以 Var_D0 的内容进行查找，将查找结果存储到 Var_W0 中的程序。

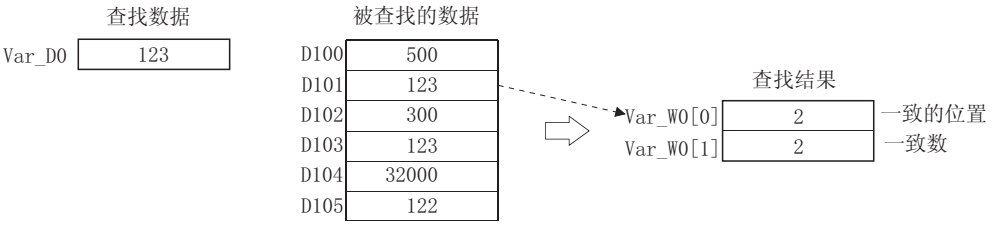
[结构体梯形图]



[ST]

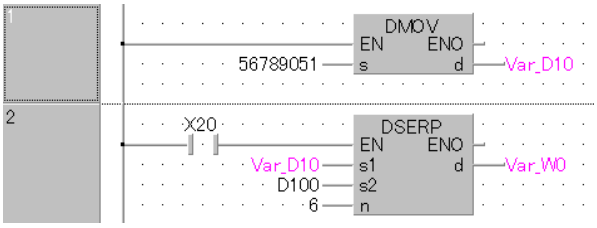
```
Var_D0:=123;
SERP(X20,Var_D0,D100,6,Var_W0);
```

[动作]



(2) 以下为 X20 变为 ON 时，对 D100 ～ D111 以 Var_D10 的内容进行查找，将查找结果存储到 Var_W0 中的程序。

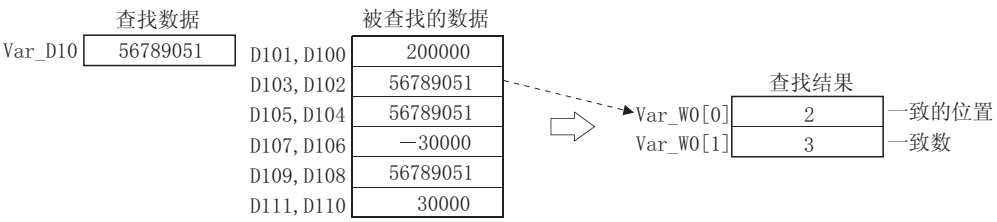
[结构体梯形图]



[ST]

```
Var_D10:=56789051;
DSERP(X20,Var_D10,D100,6,Var_W0);
```

[动作]



7.5.2 16 位 /32 位数据的位校验

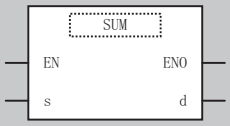
SUM, DSUM

Basic | High performance | Universal | L CPU

SUM(P)
DSUM(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= SUM (EN, s, d);

□ 中放入下述指令。

SUM	SUMP
DSUM	DSUMP

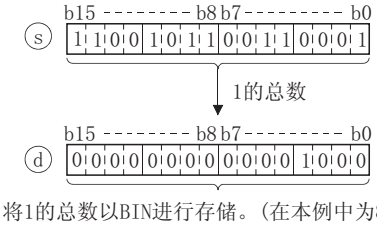
输入自变量， EN: 执行条件 : 位
s: 对处于 1 状态的位的总数进行计数得的数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 存储位的总数的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ		○						○	-
ⓓ		○						-	-

功 能

SUM(P)

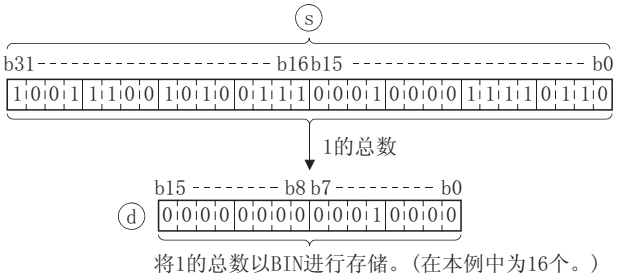
在ⓢ中指定的软元件的 16 位数据内，将处于 1 状态的位的总数存储到ⓓ中指定的软元件中。



将1的总数以BIN进行存储。(在本例中为8个。)

DSUM(P)

在ⓢ中指定的软元件的 32 位数据内，将处于 1 状态的位的总数存储到ⓓ中指定的软元件中。



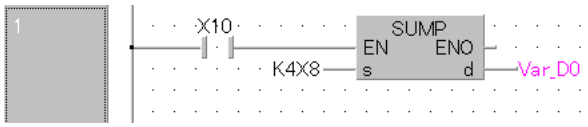
出 错

不存在 SUM(P)、,DSUM(P) 指令相关的运算出错。

程序示例

(1) 以下为 X10 变为 ON 时，对 X8 ～ X17 中处于 ON 状态的位数存储到 Var_D0 中的程序。

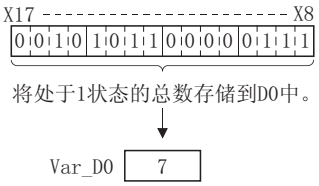
[结构体梯形图]



[ST]

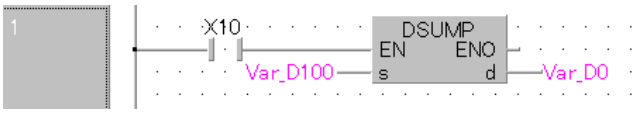
```
SUMP(X10,K4X8,Var_D0);
```

[动作]



(2) 以下为 X10 变为 ON 时，将 Var_D100 中处于 ON 的位数存储到 Var_D0 中的程序。

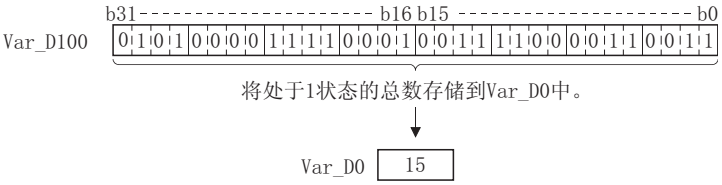
[结构体梯形图]



[ST]

```
DSUMP(X10,Var_D100,Var_D0);
```

[动作]



7.5.3 8→256 位解码

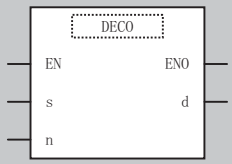
DECO

Basic | High performance | Universal | L CPU

DECO(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= DECO (EN, s, n, d);

□ 中放入下述指令。

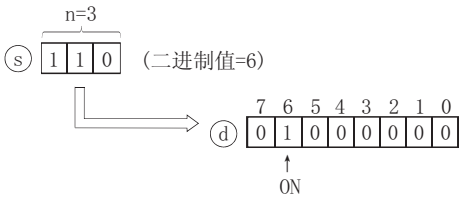
DECO　　DECOP

输入自变量，EN:　执行条件　：位
　　s:　解码数据　：ANY_SIMPLE
　　n:　有效位长（1～8），0 时无处理　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　解码结果　：ANY_SIMPLE

设置数据	内部软元件 位	内部软元件 字	R, ZR	J□\□ 位	J□\□ 字	U□\G□	Zn	常数 K, H	其它
ⓢ	○			○				○	-
n	○			○				○	-
ⓓ	○			-				-	-

功 能

(1) 将ⓢ的低 n 位中指定的二进制值对应的ⓓ的位位置置为 ON。



(2) n 的可指定范围为 1～8。

(3) n=0 时执行无处理，ⓓ中指定的软元件的内容不变化。

(4) 位软元件作为 1 位处理，字软元件作为 16 位处理。

出 错

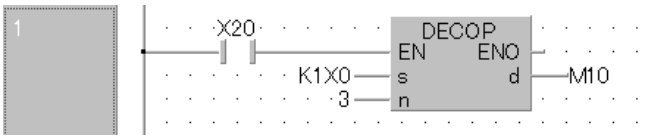
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 超出了 0 ～ 8 的范围时。 （出错代码： 4100）
- ⓓ 算起的 2^n 位的范围超出了相应软元件的范围时。 （出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，将 X0 算起的 3 位进行解码后，将结果存储到 M10 以后的程序

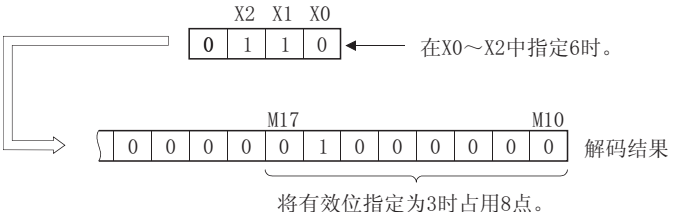
[结构体梯形图]



[ST]

```
DECOP(X20,K1X0,3,M10);
```

[动作]



7.5.4 256 → 8 位编码

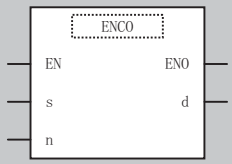
ENCO

Basic | High performance | Universal | L CPU

ENCO(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ENCO (EN, s, n, d);

□ 中放入下述指令。

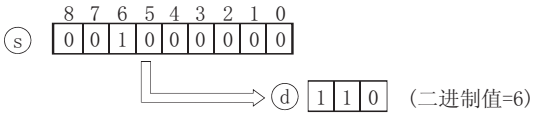
ENCO　　ENCOP

输入自变量，EN:　执行条件　：位
　　s:　编码数据　：ANY_SIMPLE
　　n:　有效位长（1 ～ 8），0 时无处理　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　编码结果　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○			－				－	－
n	○			○				○	－
ⓓ	○			○				－	－

功 能

(1) 从ⓢ的 2^n 位的数据开始，将处于 1 状态的位所对应的二进制值存储到ⓓ中。



(2) n 的可指定范围为 1 ～ 8。

(3) n=0 时执行无处理，ⓓ中指定的软元件的内容不变化。

(4) 位软元件作为 1 位处理，字软元件作为 16 位处理。

(5) 多个位为 1 时以高位的位位置进行处理。

出 错

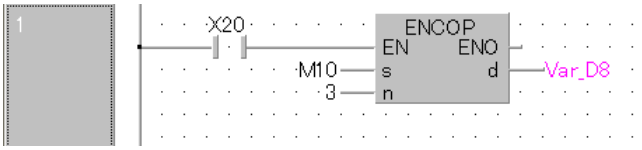
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- n 超出了 0 ～ 8 的范围时。（出错代码： 4100）
- ⓢ 算起的 2^n 位的范围超出了相应软元件的范围时。（出错代码： 4101）
- ⓢ 算起的 2^n 位的数据全部为 0 时。（出错代码： 4100）

程序示例

以下为 X20 变为 ON 时，对 M10 算起的 3 位进行编码后，将结果存储到 Var_D8 中的程序。

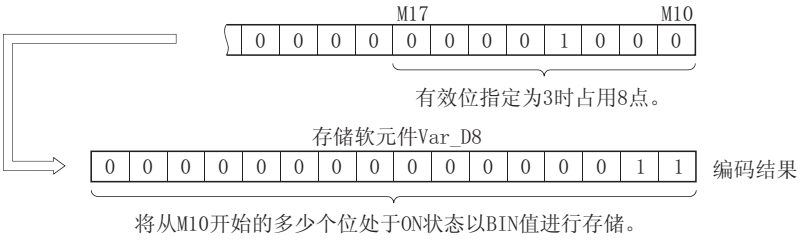
[结构体梯形图]



[ST]

```
ENCOP(X20,M10,3,Var_D8);
```

[动作]



7.5.5 7段解码

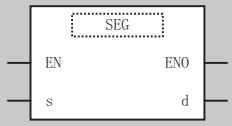
SEG

Basic | High performance | Universal | L CPU

SEG(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SEG (EN, s, d);

[ ] 中放入下述指令。

SEG　　SEGP

输入自变量，EN:　执行条件　：位
　　s:　解码数据或存储解码数据的软元件的起始编号　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　存储解码结果的软元件的起始编号　：ANY16

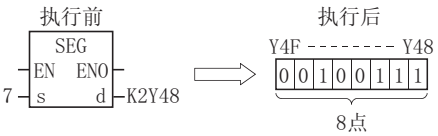
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○				○	-
ⓓ				○				-	-

功 能

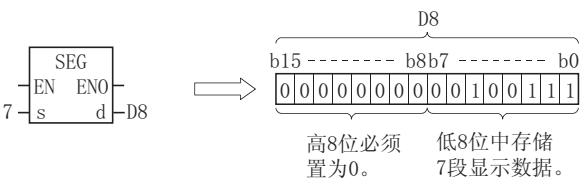
(1) 将ⓢ的低4位中指定的0～F的数据解码为7段显示的数据后，存储到ⓓ中。

(2) 位软元件时的ⓓ表示存储7段显示数据的软元件的起始编号，在字软元件中表示存储的软元件编号。

位软元件



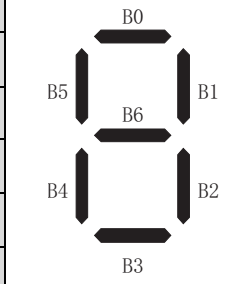
字软元件



出 错

不存在 SEG(P) 指令相关的运算出错。

7 段解码表

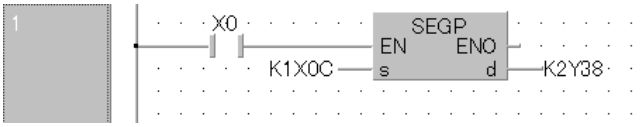
ⓢ		7 段的构成	ⓓ								显示数据
16 进制数	位模式		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	b
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	d
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

位软元件的起始
字软元件的最低位

程序示例

以下为 X0 变为 ON 时，将 XC ～ XF 的数据转换为 7 段显示数据后，输出到 Y38 ～ Y3F 中的程序。

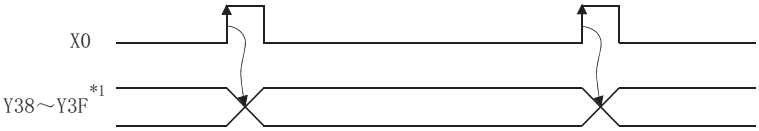
[结构体梯形图]



[ST]

```
SEGP(X0,K1X0C,K2Y38);
```

[时序图]



*1: Y38～Y3F在下一个数据被输出之前不变化。

7.5.6 16 位数据的 4 位分离

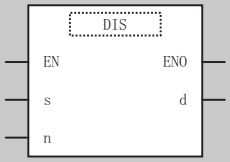
DIS

Basic | High performance | Universal | L CPU

DIS(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= DIS (EN, s, n, d);

[ ] 中放入下述指令。

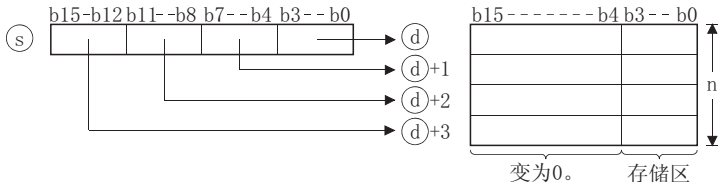
DIS DISP

输入自变量， EN: 执行条件 : 位
s: 存储分离数据的软元件的起始编号 : ANY16
n: 分离数 (1 ～ 4)，0 时无处理 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储被分离数据的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
n	○	○		○					-
ⓓ	-	○		-					-

功 能

(1) 将ⓢ中指定的 16 位数据的低 n 位数 (1 位数 4 位) 的数据，存储到ⓓ中指定的软元件算起 n 点的低 4 位中。



(2) ⓓ中指定的软元件算起 n 点的高 12 位将变为 0。

(3) n 中可指定的范围为 1 ～ 4。

(4) n=0 时执行无处理，ⓓ的软元件算起 n 点的内容不变化。

出 错

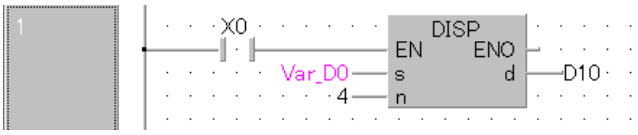
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 算起 n 点的范围超出了相应软元件的范围时。 （出错代码： 4101）
- n 超出了 0 ～ 4 的范围时。 （出错代码： 4100）

程序示例

以下为 X0 变为 ON 时，将从 Var_D0 开始的 16 位数据以 4 位为单位进行分离后，存储到 D10 ～ D13 中的程序。

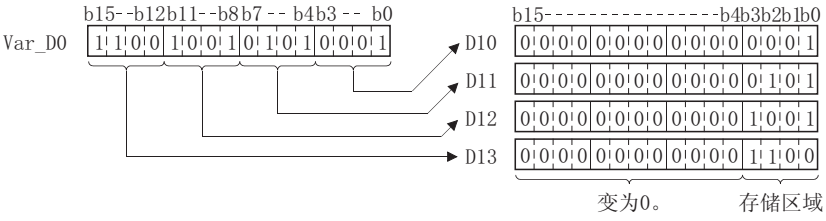
[结构体梯形图]



[ST]

```
DISP(X0,Var_D0,4,D10);
```

[动作]



7.5.7 16 位数据的 4 位合并

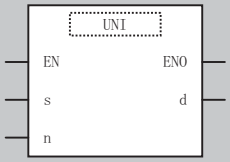
UNI

Basic | High performance | Universal | L CPU

UNI(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= UNI (EN, s, n, d);

[　　] 中放入下述指令。

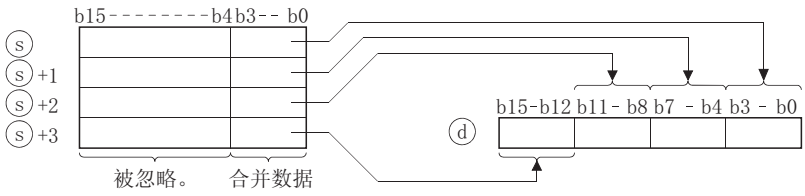
UNI　　UNIP

输入自变量，EN:　执行条件　：位
　　s:　存储合并数据的软元件的起始编号　：ANY16
　　n:　合并数（1 ～ 4），0 时无处理　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　存储被合并的数据的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	–	○		–				–	–
n	○	○		○				○	–
ⓓ	○	○		○				–	–

功 能

(1) 将ⓢ中指定的软元件算起 n 点的 16 位数据的低位 4 位，与ⓓ中指定的 16 位软元件进行合并。



(2) ⓓ中指定的软元件的高位 (4-n) 的位数的位将变为 0。

(3) n 中可指定的范围为 1 ～ 4。

(4) n=0 时执行无处理，ⓓ软元件的内容不变化。

出 错

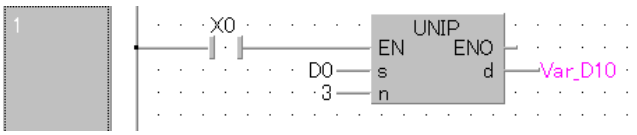
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ算起 n 点的范围超出了相应软元件的范围时。（出错代码： 4101）
- n 超出了 0 ～ 4 的范围时。（出错代码： 4100）

程序示例

以下 X0 变为 ON 时，将 D0 ～ D2 的低 4 位进行合并后，存储到 Var_D10 中的程序。

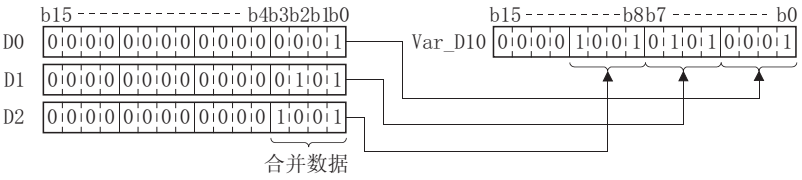
[结构体梯形图]



[ST]

```
UNIP(X0,D0,3,Var_D10);
```

[动作]



7.5.8 任意数据的位分离、合并

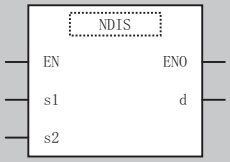
NDIS, NUNI

Basic | High performance | Universal | L CPU

NDIS(P)
NUNI(P)

P: 执行条件 : ⌐

结构体梯形图



ST

ENO:= NDIS (EN, s1, s2, d);

[ ] 中放入下述指令。

NDIS	NDISP
NUNI	NUNIP

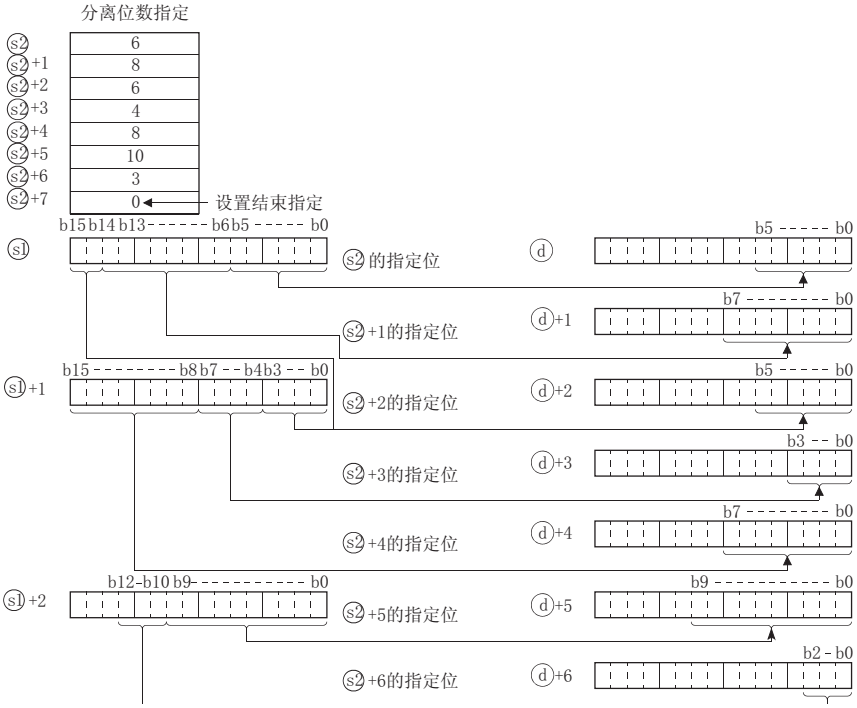
输入自变量， EN: 执行条件 : 位
s1: 存储分离、合并数据的软元件的起始编号 : ANY16
s2: 存储分离、合并单位的软元件的起始编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储被分离、合并的数据的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
⑤1	-	○		-					
⑤2	-	○		-					
ⓓ	-	○		-					

功 能

NDIS(P)

(1) 将㉑中指定的软元件编号以后中存储的数据的各个位，以㉒中指定的位为单位进行分离后，存储到ⓓ中指定的软元件编号以后。



(2) ㉒中指定的分离位数的可指定范围为 1 ～ 16 位。

(3) 将㉒中指定的软元件编号开始至存储了 0 的软元件编号为止作为分离位数处理。

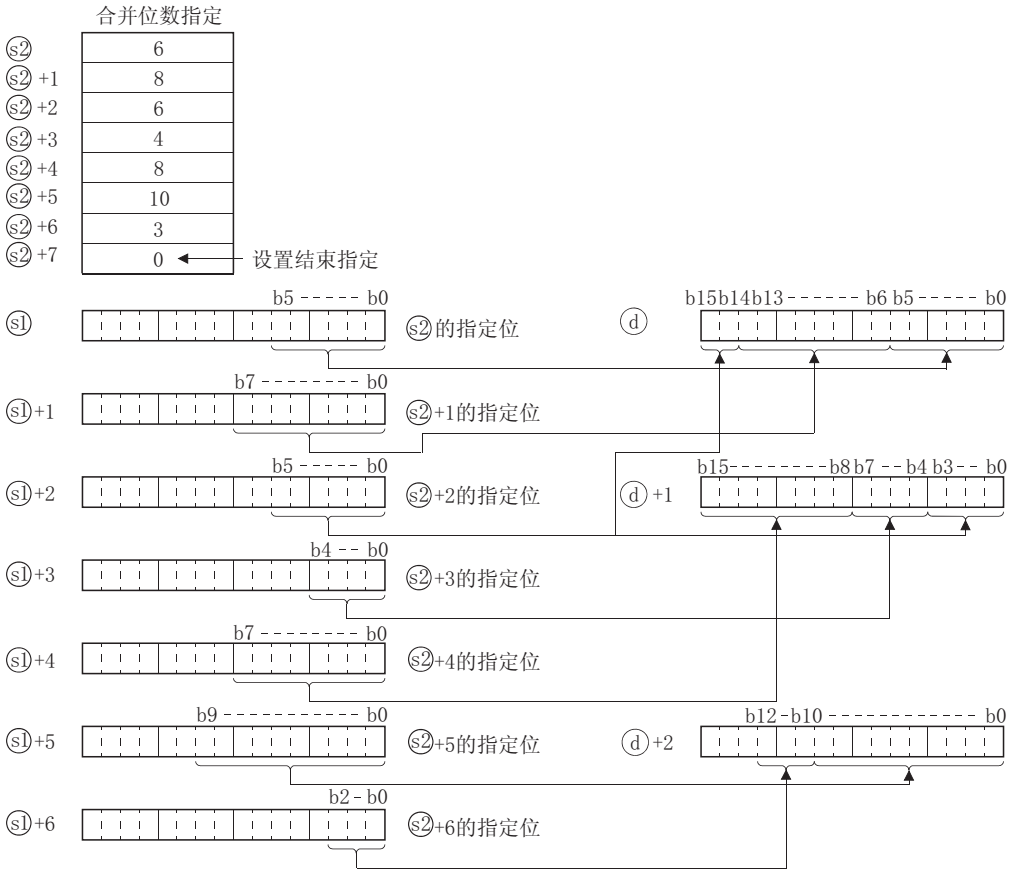
(4) 分离数据的软元件范围（㉑～㉑的结束范围）与存储分离后数据的软元件范围（ⓓ～ⓓ结束范围）不要重叠。

如果重叠则可能无法得到正确的运算结果。

(5) 应避免使㉑，㉒，ⓓ中指定的软元件编号重复。如果重复，将无法进行正确的运算。

NUNI(P)

(1) 将ⓢ1 中指定的软元件编号以后中存储的数据的各个位，以ⓢ2 中指定的位进行合并后，存储到ⓓ中指定的软元件编号以后。



(2) ⓢ2 中指定的合并位数的可指定范围为 1 ～ 16 位。

(3) 将ⓢ2 中指定的软元件编号开始至存储了 0 的软元件编号为止作为合并位数处理。

(4) 合并数据的软元件范围（ⓢ1 ～ⓢ1 的结束范围）与存储合并后数据的软元件范围（ⓓ ～ⓓ 的结束范围）不要重叠。
如果重叠则可能无法得到正确的运算结果。

(5) 应避免使ⓢ1，ⓢ2，ⓓ中指定的软元件编号重复。如果重复，将无法进行正确的运算。

出 错

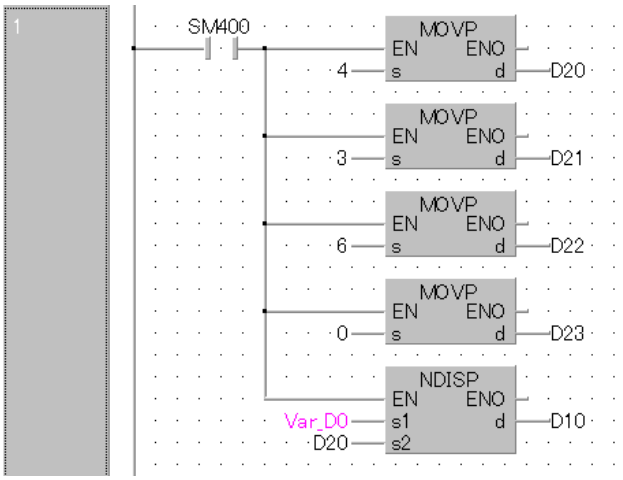
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 由于ⓢ2 中指定的分离、合并位数指定，ⓢ1 或ⓓ中指定的软元件的使用范围超出了各自的软元件的最终软元件编号时。（出错代码： 4101）
- ⓢ2 中指定的分离、合并位数指定超出了 1 ～ 16 位的设置范围时。（出错代码： 4100）

程序示例

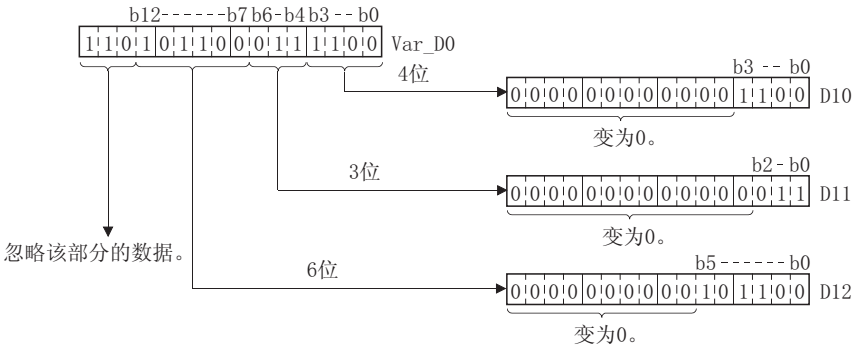
(1) 以下为从 Var_D0 的数据的低位开始分别分离为 4、3、6 位后，存储到 D10 ～ D12 中的程序。

[结构体梯形图]



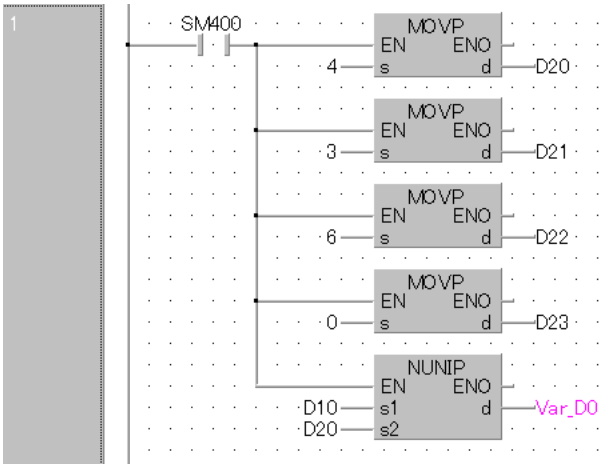

```
[ST]
IF SM400 THEN
     MOVP(TRUE,4,D20);
     MOVP(TRUE,3,D21);
     MOVP(TRUE,6,D22);
     MOVP(TRUE,0,D23);
     NDISP(TRUE,Var_D0,D20,D10);
END_IF;
```

[动作]



(2) 以下为将 D10 的数据的低 4 位、D11 的数据的低 3 位与 D12 的数据的低 6 位进行合并后，存储到 Var_D0 中的程序。

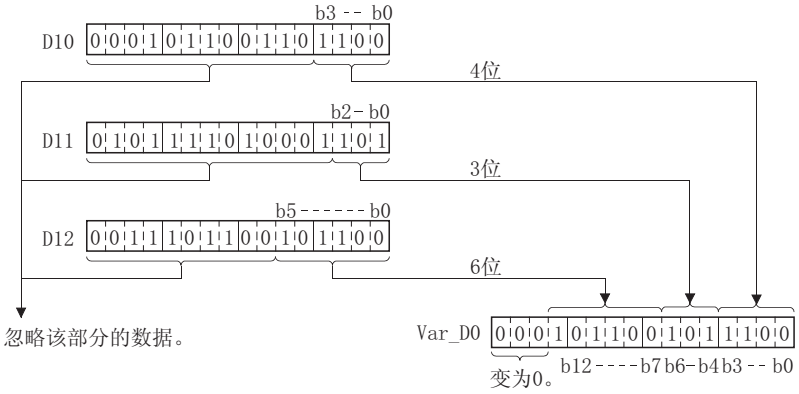
[结构体梯形图]



[ST]

```
IF SM400 THEN
    MOVP(TRUE,4,D20);
    MOVP(TRUE,3,D21);
    MOVP(TRUE,6,D22);
    MOVP(TRUE,0,D23);
    NUNIP(TRUE,D10,D20,Var_D0);
END_IF;
```

[动作]



7.5.9 字节单位数据分离、合并

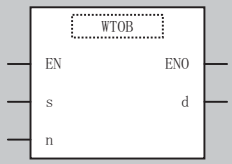
WTOB, BTOW

Basic | High performance | Universal | L CPU

WTOB(P)
BTOW(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= WTOB (EN, s, n, d);
```

[ ] 中放入下述指令。

WTOB	WTOBP
BTOW	BTOWP

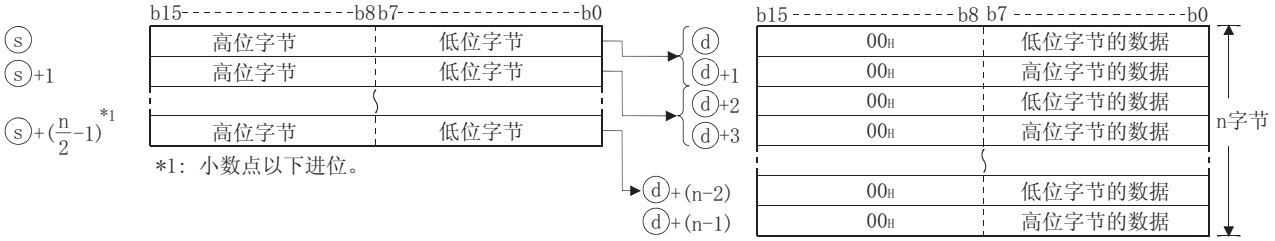
输入自变量， EN: 执行条件 : 位
s: 存储以字节单位进行分离、合并的数据的软元件的起始编号 : ANY16
n: 进行分离、合并的字节数据的个数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储以字节单位进行了分离、合并的结果的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-					-
n	○	○		○					-
ⓓ	-	○		-					-

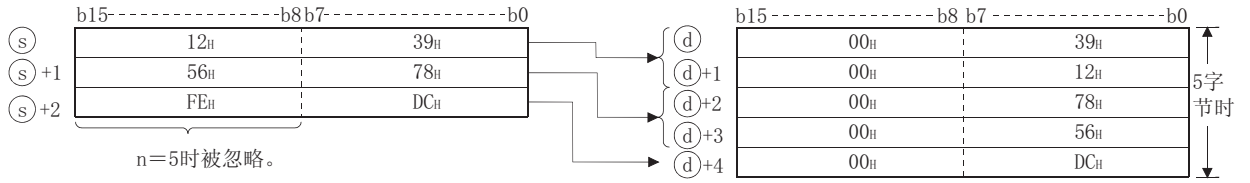
功 能

WTOB(P)

(1) 将ⓢ中指定的软元件编号以后存储的 16 位数据，以 n 字节进行分离后，存储到ⓓ中指定的软元件编号以后。



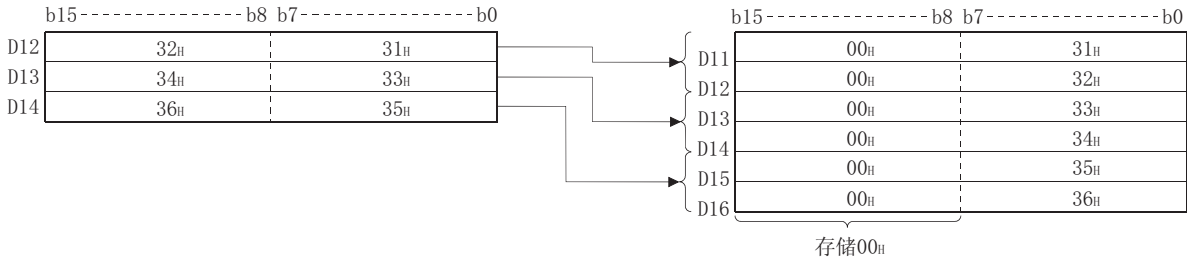
例如 n=5 时，将ⓢ～(ⓢ+2)的低 8 位为止的数据存储到ⓓ～(ⓓ+4)中。



(2) 通过在 n 中设置字节数，ⓢ中指定的 16 位数据的范围以及存储ⓓ中指定的字节数据的软元件的范围将被自动确定。

(3) n 中指定的字节数为 0 时，将不进行处理。

(4) ⓓ中指定的字节数据存储软元件的高 8 位中将自动地存储 00H。

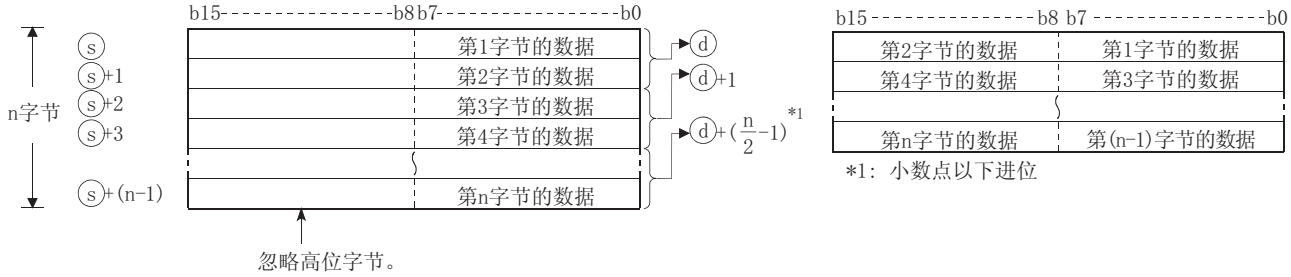


BTOW(P)

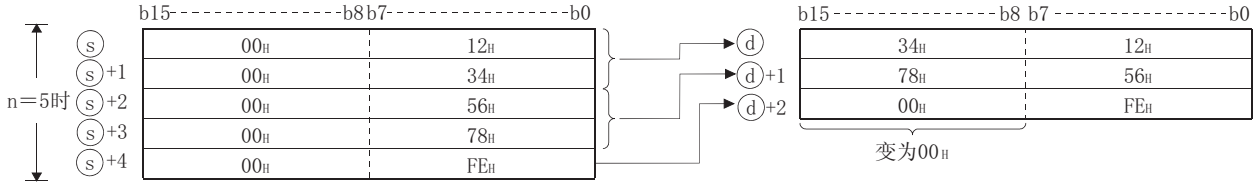
(1) 将ⓢ中指定的软元件编号以后存储的 n 字的 16 位数据的低 8 位合并为字单位后，存储到ⓓ中指定的软元件编号以后。

ⓢ中指定的软元件编号以后存储的 n 字的数据的高 8 位将被忽略。

此外，n 为奇数的情况下，在存储了第 n 字节的数据的软元件的高 8 位中将存储 0。



例如 n=5 时，将ⓢ～(ⓢ+4)的低 8 位的数据进行合并后存储到ⓓ～(ⓓ+2)中。



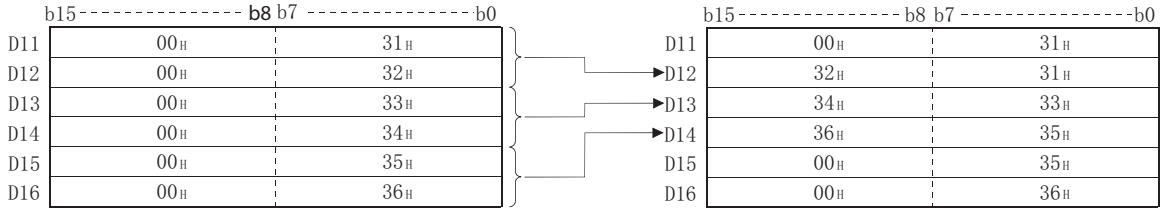
(2) 通过在 n 中设置字节数，ⓢ中指定的字节数据的范围以及ⓓ中指定的合并数据存储软元件的范围将被自动确定。

(3) n 中指定的字节数为 0 时，将不进行处理。

(4) ⓢ中指定的字节数据存储软元件的高 8 位将被忽略，低 8 位将成为对象。

(5) 即使存储合并数据的软元件范围(ⓢ～ⓢ+(n-1))与存储合并后数据的软元件范围(ⓓ～ⓓ+(n/2-1))相重叠，也将正常进行处理。

例如将 D11 ～ D16 的低 8 位存储到 D12 ～ D14 中的情况如下图所示。



出 错

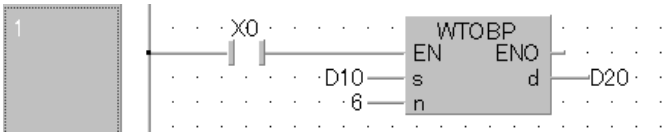
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件编号以后，n 中指定的字节数的范围超出了相应软元件的范围时。
  (出错代码： 4101)
- ⓓ 中指定的软元件编号以后，n 中指定的字节数的范围超出了相应软元件的范围时。
  (出错代码： 4101)

程序示例

(1) 以下为将 X0 置为 ON 时，将 D10 ～ D12 的数据分离为字节单位后，存储到 D20 ～ D25 中的程序。

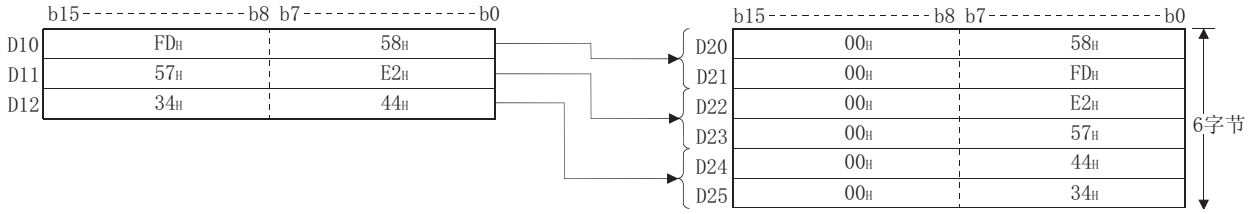
[结构体梯形图]



[ST]

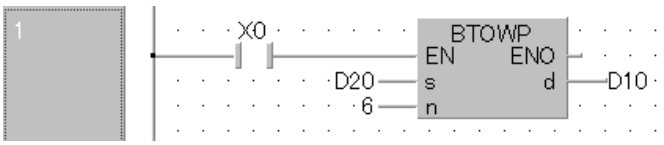
```
WTOBP(X0,D10,6,D20);
```

[动作]



(2) 以下为将 X0 置为 ON 时，将 D20 ～ D25 的低 8 位数据进行合并后，存储到 D10 ～ D12 中的程序。

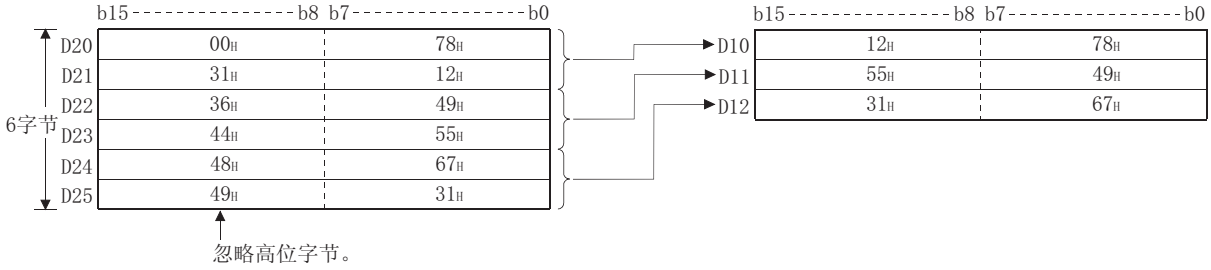
[结构体梯形图]



[ST]

```
BTOWP(X0,D20,6,D10);
```

[动作]



7.5.10 16 位 /32 位数据最大值查找

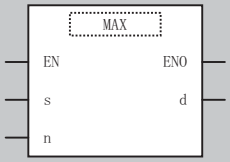
MAX, DMAX

Basic | High performance | Universal | L CPU

MAX(P)
DMAX(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= MAX (EN, s, n, d);

[ ] 中放入下述指令。

MAX	MAXP
DMAX	DMAXP

输入自变量， EN: 执行条件 : 位
s: 查找最大值的软元件的起始编号 : ANY16/32
n: 查找数据数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 最大值的查找结果 : ANY16/32

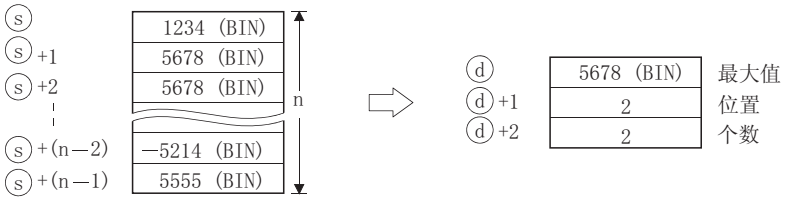
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-					-
n	○	○		○					-
ⓓ	-	○		-					-

功 能

MAX(P)

从ⓢ中指定的软元件开始，从 n 点的 16 位 BIN 数据中查找最大值后，将最大值存储到ⓓ中指定的软元件中。

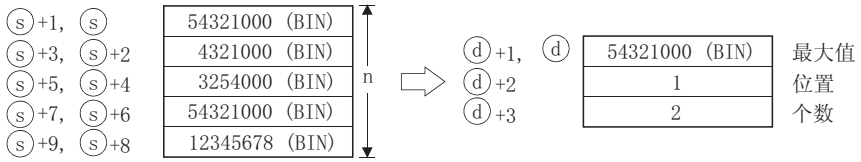
从ⓢ中指定的软元件开始查找，将存储最初检测到的最大值的软元件编号至ⓢ的点数存储到ⓓ+1 中，将最大值的个数存储到ⓓ+2 中。



DMAX(P)

从ⓢ中指定的软元件开始，从 n 点的 32 位 BIN 数据中查找最大值，将最大值存储到ⓓ，ⓓ+1 中指定的软元件中。

从ⓢ中指定的软元件开始查找，将存储最初检测到的最大值的软元件编号至ⓢ的点数存储到ⓓ+2 中，将最大值的个数存储到ⓓ+3 中。



出 错

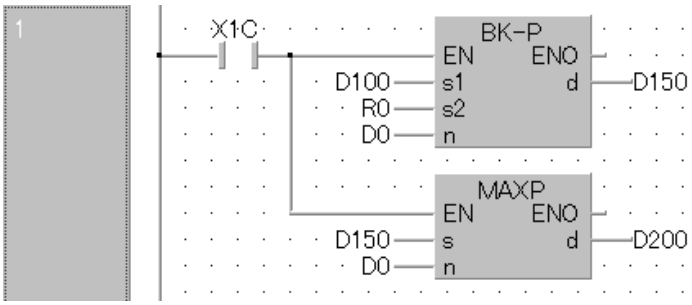
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ的软元件算起 n 点的范围超出了相应软元件时。 （出错代码： 4101）
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

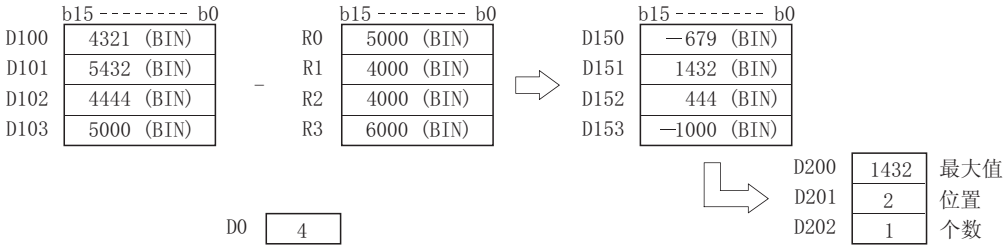
(1) 以下为 X1C 变为 ON 时，将 D100 ～ D103 中存储的值的数据，与 R0 ～ R3 中存储的值的数据进行减法运算，从该结果中查找最大值后，存储到 D200 ～ D202 中的程序。

[结构体梯形图]



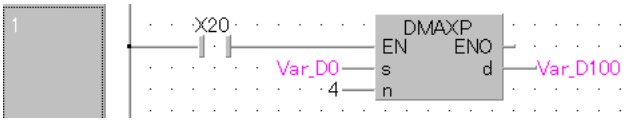

```
[ST]
IF X1C THEN
     D0:=4;
     D150:=D100-R0;
     D151:=D101-R1;
     D152:=D102-R2;
     D153:=D103-R3;
     MAXP(TRUE,D150,D0,D200);
END_IF;
```

[动作]



(2) 以下为 X20 变为 ON 时，从 Var_D0 的 32 位数据中查找最大值后，存储到 Var_D100 中的程序。

[结构体梯形图]



[ST]

```
DMAXP(X20,Var_D0,4,Var_D100);
```

[动作]

7.5.11 16 位 /32 位数据最小值查找

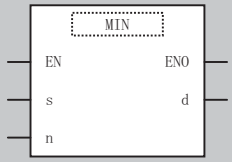
MIN, DMIN

Basic | High performance | Universal | L CPU

MIN(P)
DMIN(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= MIN (EN, s, n, d);

□ 中放入下述指令。

MIN	MINP
DMIN	DMINP

输入自变量， EN: 执行条件 : 位
s: 查找最小值的软元件的起始编号 : ANY16/32
n: 查找数据数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 最小值的查找结果 : ANY16/32

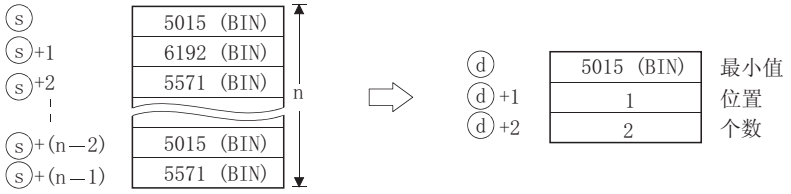
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-					-
n	○	○		○					-
ⓓ	-	○		-					-

功 能

MIN(P)

从ⓢ中指定的软元件开始，从 n 点的 16 位 BIN 数据中查找最小值后，将最小值存储到ⓓ中指定的软元件中。

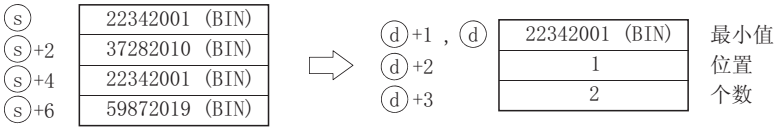
从ⓢ中指定的软元件开始查找，将存储最初检测到的最小值的软元件编号至ⓢ的点数存储到ⓓ+1 中，将最小值的个数存储到ⓓ+2 中。



DMIN(P)

从ⓢ中指定的软元件开始，从 n 点的 32 位 BIN 数据中查找最小值后，将最小值存储到ⓓ，ⓓ+1 中指定的软元件中。

从ⓢ中指定的软元件开始查找，将存储最初检测到的最小值的软元件编号至ⓢ的点数存储到ⓓ+2 中，将最小值的个数存储到ⓓ+3 中。



出 错

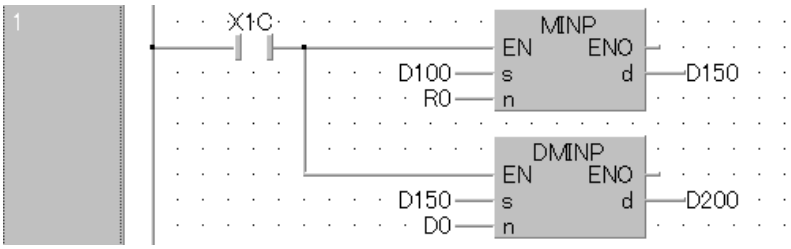
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ的软元件算起 n 点的范围超出了相应软元件时。 (出错代码： 4101)
- ⓓ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

程序示例

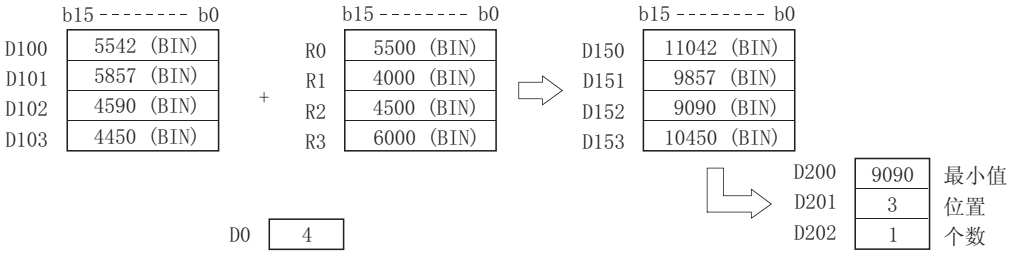
(1) 以下为 X1C 变为 ON 时，将 D100 ～ D103 中存储的值的数据，与 R0 ～ R3 中存储的值的数据进行加法运算，从该结果中查找最小值后，存储到 D200 ～ D202 中的程序。

[结构体梯形图]



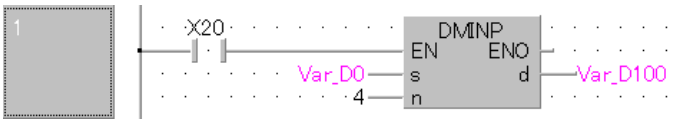

```
[ST]
IF X1C THEN
     D0:=4;
     D150:=D100+R0;
     D151:=D101+R1;
     D152:=D102+R2;
     D153:=D103+R3;
     MINP(TRUE,D150,D0,D200);
END_IF;
```

[动作]



(2) 以下为 X20 变为 ON 时，从 Var_D0 的 32 位数据中查找最小值后，存储到 Var_D100 中的程序。

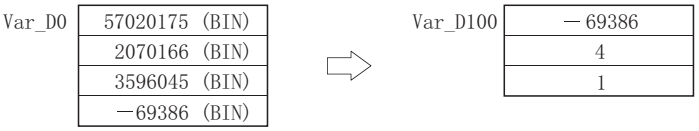
[结构体梯形图]



[ST]

```
DMINP(X20,Var_D0,4,Var_D100);
```

[动作]



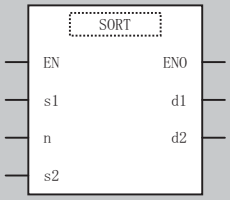
7.5.12 16 位 /32 位数据排序

SORT, DSORT

Basic | High performance | Universal | L CPU

SORT
DSORT

结构体梯形图



ST

```
ENO:=  SORT  (EN, s1, n, s2, d1, d2);
```

[ ] 中放入下述指令。

SORT
DSORT

输入自变量， EN: 执行条件 : 位
s1: 排序的表 : ANY16/32
s2: 1 次执行中进行比较的数据数 : ANY16
n: 排序数据数 : ANY16
输出自变量， ENO: 执行结果 : 位
d1: 排序结束时使其 ON 的位软元件编号 : 位
d2: 系统使用软元件 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1	–	○		–					–
⑤2	○	○		○					–
n	○	○		○					–
⑪1	○	–		–					–
⑫2	–	○		–					–

功 能

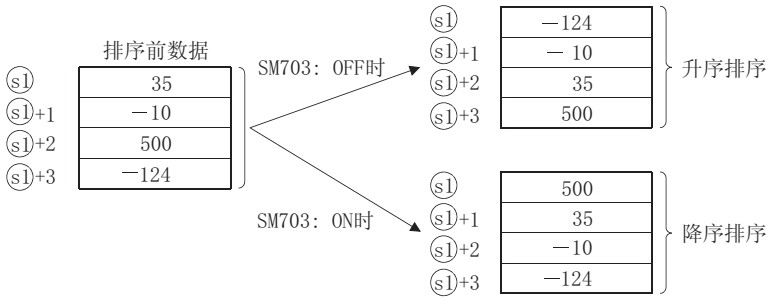
SORT

(1) 将㉑算起 n 点的 BIN 16 位数据进行升序 / 降序的排序（排列）。

排序是通过 SM703 的 ON/OFF 进行指定。

• SM703 为 OFF 时 : 升序排序

• SM703 为 ON 时 : 降序排序



(2) 通过 SORT 指令进行的排序需要数个扫描。

至执行结束为止的扫描次数为，将排序执行结束为止的最大执行次数，用㉒中指定的 1 次执行中比较的数据数相除后的值。（小数点以下进位。）

㉒的值较大时，至排序结束为止的扫描次数将变少，扫描时间将延长。

(3) 至排序执行结束为止的最大执行次数可通过下式进行计算。

至执行结束为止的最大执行次数 =(n) × (n-1) ÷ 2[次]

例

n=10 的情况下，需要 10 × (10-1) ÷ 2=45[次]。

此时如果㉒=2，至排序结束为止变为 45 ÷ 2=22.5→23[扫描]。

(4) 对于㉓中指定的软元件（结束软元件），开始执行 SORT 指令时将变为 OFF，排序结束时将变为 ON。

排序结束后，㉓中指定的软元件将保持为 ON 状态不变，因此应根据需要由用户将其置为 OFF。

(5) 对于㉜中指定的软元件算起的 2 点，在执行 SORT 指令时为系统所使用。

对于㉜中指定的软元件算起的 2 点，用户不要对起进行变更。

如果进行了变更，有可能变为出错状态。 （出错代码 : 4100）

(6) 如果在排序的执行过程中对 n 进行了变更，将以变更后的排序数据数进行排序。

(7) 在排序的执行过程中如果将执行指令置为 OFF，排序将被中断。

再次将执行指令置为 ON 时，排序将从最初开始重新进行。

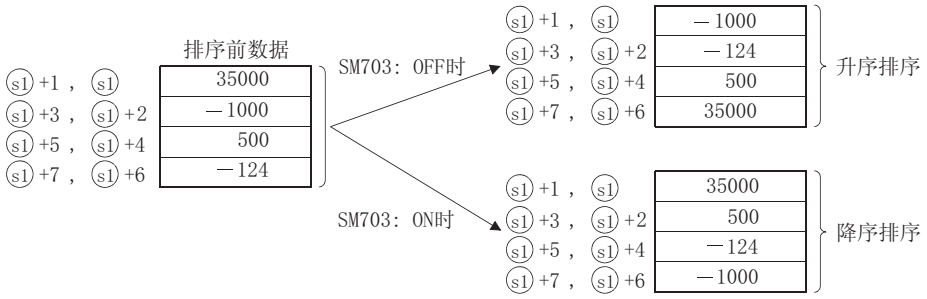
(8) 排序执行结束后，连续进行下一个排序的情况下，需要将执行指令置为 OFF 后，再次将执行指令置为 ON。

DSORT

(1) 将㉛算起 n 点的 BIN 32 位数据进行升序 / 降序的排序（排列）。

排序是通过 SM703 的 ON/OFF 进行指定。

- SM703 为 OFF 时 : 升序排序
- SM703 为 ON 时 : 降序排序



(2) 通过 DSORT 指令进行的排序需要数个扫描。

至执行结束为止的扫描次数为，将排序执行结束为止的最大执行次数，用㉜中指定的 1 次执行中比较的数据数相除后的值。（小数点以下进位。）

㉜的值较大时，至排序结束为止的扫描次数将变少，扫描时间将延长。

(3) 至排序执行结束为止的最大执行次数可通过下式进行计算。

至执行结束为止的最大执行次数 =(n) × (n-1) ÷ 2 [次]

例

n=10 的情况下，需要 10 × (10-1) ÷ 2=45[次] 修。
此时如果 S2=2，至排序结束为止变为 45 ÷ 2=22.5→23[扫描]。

(4) 对于㊱中指定的软元件（结束软元件），开始执行 DSORT 指令时将变为 OFF，排序结束时将变为 ON。

排序结束后，㊱中指定的软元件将保持为 ON 状态不变，因此应根据需要由用户将其置为 OFF。

(5) 对于㊲中指定的软元件算起的 2 点，在执行 DSORT 指令时为系统所使用。

对于㊲中指定的软元件算起的 2 点，用户不要对起进行变更。

如果进行了变更，有可能变为出错状态。 （出错代码 : 4100）

(6) 如果在排序的执行过程中对 n 进行了变更，将以变更后的排序数据数进行排序。

(7) 在排序的执行过程中如果将执行指令置为 OFF，排序将被中断。
再次将执行指令置为 ON 时，排序将从最初开始重新进行。

(8) 排序执行结束后，连续进行下一个排序的情况下，需要将执行指令置为 OFF 后，再次将执行指令置为 ON。

出 错

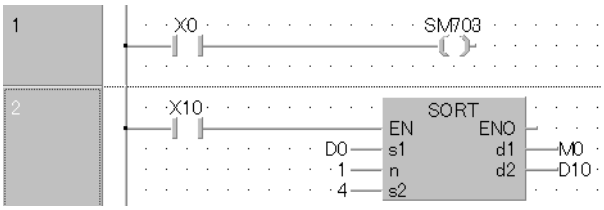
在以下情况下将变为运算出错状态，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- SORT 指令中㉛算起 n 点的范围超出了相应软元件范围时。（出错代码：4101）
- DSORT 指令中㉛算起 2×n 点的范围超出了相应软元件范围时。（出错代码：4101）
- ㉛算起 n 点÷2×n 点的软元件范围与㉜算起 2 点的软元件范围重叠时。（出错代码：4101）
- ㉜为 0 或负数时。（出错代码：4100）

程序示例

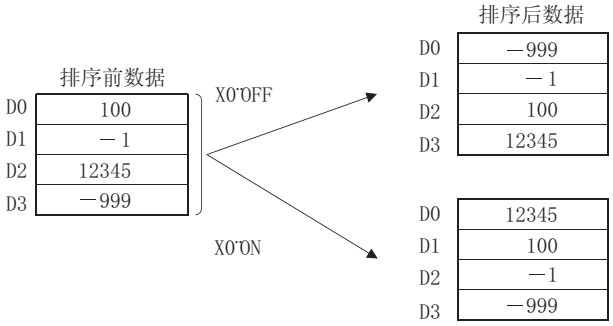
(1) 以下为 X10 变为 ON 时，将 D0 算起 4 点的 BIN16 位数据进行升序 / 降序排序的程序。

[结构体梯形图]



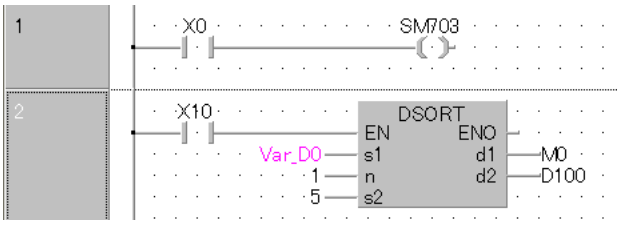

```
[ST]
OUT(X0,SM703);
SORT(X10,D0,1,4,M0,D10);
```

[动作]



(2) 以下为 X10 变为 ON 时，将 D0 算起 20 点的 BIN32 位数据进行升序 / 降序排序的程序。

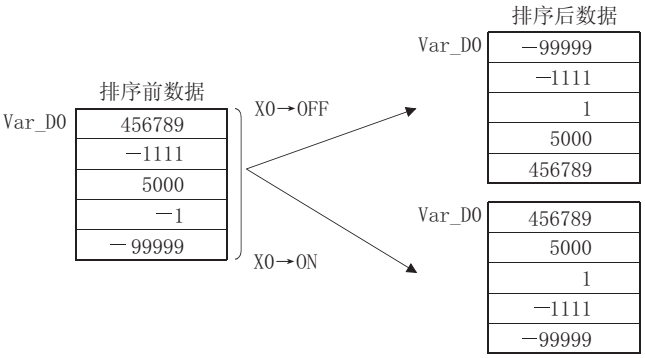
[结构体梯形图]



[ST]

```
OUT(X0,SM703);
DSORT(X10,Var_D0,1,5,M0,D100);
```

[动作]



7.5.13 16 位数据合计值计算

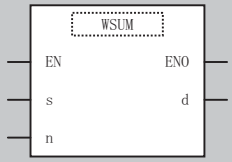
WSUM

Basic | High performance | Universal | L CPU

WSUM(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= WSUM (EN, s, n, d);

[　　] 中放入下述指令。

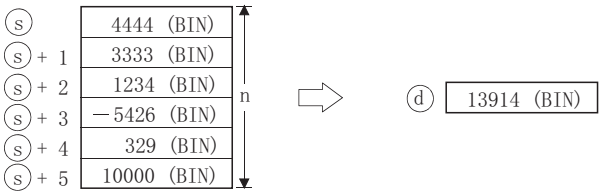
WSUM　　WSUMP

输入自变量，EN:　执行条件　：位
　　s:　存储进行合计值计算的数据的软元件的起始编号　：ANY16
　　n:　数据个数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　合计值　：ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	－	○		－				－	－
n	○	○		○				○	－
ⓓ	○	○		○				－	－

功 能

从ⓢ中指定的软元件开始，将 n 点的 16 位 BIN 数据全部进行加法运算后，存储到ⓓ中指定的软元件中。



出 错

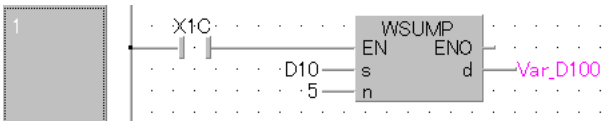
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ的软元件算起 n 点的范围超出了相应软元件时。　（出错代码： 4101）

程序示例

以下为 X1C 变为 ON 时，对 D10 ～ D14 的 16 位 BIN 数据进行加法运算后，存储到 Var_D100 中的程序。

[结构体梯形图]



[ST]

```
WSUMP(X1C,D10,5,Var_D100);
```

[动作]

D10	4500 (BIN)
D11	2500 (BIN)
D12	−3276 (BIN)
D13	6780 (BIN)
D14	4444 (BIN)

⇨ Var_D100 | 14948 (BIN)

7.5.14 32 位数据合计值计算

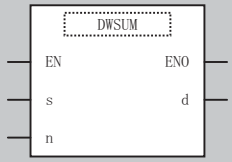
DWSUM

Basic | High performance | Universal | L CPU

DWSUM(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= DWSUM (EN, s, n, d);
```

[　　] 中放入下述指令。

DWSUM　　DWSUMP

输入自变量，EN：　执行条件　：位
　　s：　进行合计值计算的数据　：ANY32
　　n：　数据个数　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　合计值　：ANY16 的数组 (0..3)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	－	○		－				－	－
n	○	○		○				○	－
ⓓ	○	○		－				－	－

功 能

从ⓢ中指定的软元件开始，将 n 点的 32 位 BIN 数据全部进行加法运算后，存储到ⓓ中指定的软元件算起的 4 点（4 字）中。

ⓢ+1, ⓢ	32767000 (BIN)
ⓢ+3, ⓢ+2	6000 (BIN)
ⓢ+5, ⓢ+4	35392000 (BIN)
ⓢ+7, ⓢ+6	−11870000 (BIN)
ⓢ+9, ⓢ+8	12345000 (BIN)

n

⇨ ⓓ[3] ～ ⓓ[0]　68640000 (BIN)

出 错

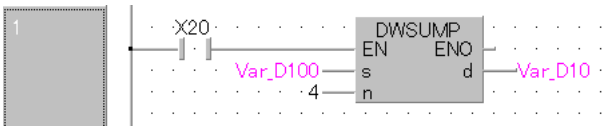
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ的软元件算起 n 点的范围超出了相应软元件时。（出错代码： 4101）
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，对 Var_D100 的 32 位 BIN 数据进行加法运算后，将结果存储到 Var_D10 中的程序。

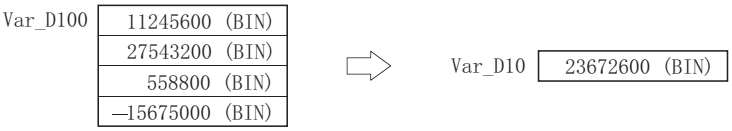
[结构体梯形图]



[ST]

```
DWSUMP(X20,Var_D100,4,Var_D10);
```

[动作]



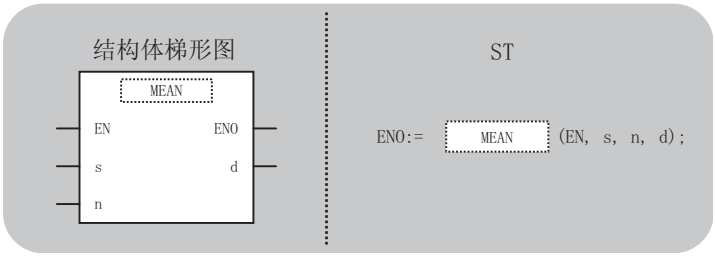
7.5.15 16 位 /32 位数据平均值计算

MEAN
DMEAN

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

MEAN(P)
DMEAN(P)

P：执行条件　　：



中放入下述指令。

MEAN	MEANP
DMEAN	DMEANP

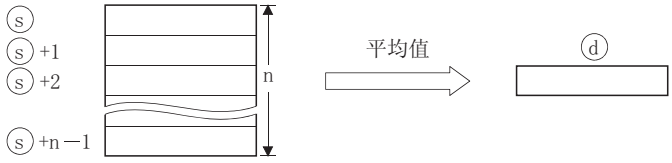
输入自变量，	EN:	执行条件	：位
	s:	存储进行平均值计算的数据的软元件的起始编号	：ANY16
	n:	数据数或存储数据数的软元件编号 （设置范围 1 ～ 32767）	：ANY16
输出自变量，	ENO:	执行结果	：位
	d:	存储平均值的软元件的起始编号	：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	－	－	○	－				－	－
n	－	○	○	○				○	－
ⓓ	－	－	○	－				－	－

功 能

MEAN(P)

(1) 将ⓢ中指定的软元件算起的 n 点（16 位 BIN 数据）的平均值进行计算后，存储到ⓓ中指定的软元件中。

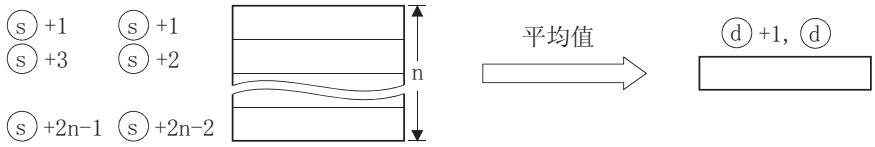


(2) 计算结果不为整数值的情况下，小数点以下将被舍去。

(3) n 中指定的值为 0 时将变为无处理。

DMEAN(P)

(1) 将ⓢ中指定的软元件算起的 n 点（32 位 BIN 数据）的平均值进行计算后，存储到ⓓ中指定的软元件中。



(2) 计算结果不为整数值的情况下，小数点以下将被舍去。

(3) n 中指定的值为 0 时将变为无处理。

出 错

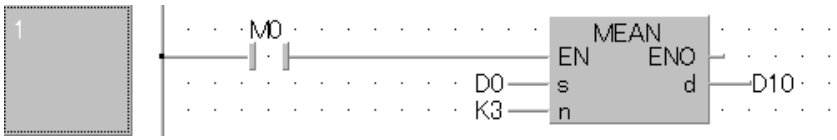
(1) 在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- n 中指定的值超出了 0 ～ 32767 的范围时。（出错代码： 4100）
- ⓢ中指定的软元件算起的 n 点超出了指定软元件的范围时。（出错代码： 4101）

程序示例

(1) 以下为 M0 变为 ON 时，将 D0 ～ D2 的 16 位数据的平均值存储到 D10 中的程序。

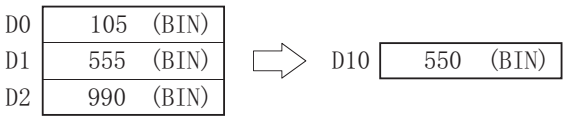
[结构体梯形图]



[ST]

```
MEAN(M0,D0,K3,D10);
```

[动作]



(2) 以下为 M0 变为 ON 时，将 D0 ～ D5 的 32 位数据的平均值存储到 D10、D11 中的程序。

[结构体梯形图]

[ST]

```
DMEAN(M0,D0,K3,D10);
```

[动作]

D1,D0	623541 (BIN)	⇨	D11,D10	2101176 (BIN)
D3,D2	4753647 (BIN)			
D5,D4	926342 (BIN)			

7.6 结构体指令

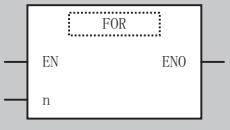
7.6.1 FOR ～ NEXT

FOR, NEXT

Basic | High performance | Universal | L CPU

FOR
NEXT

结构体梯形图



ST

不支持。

不支持。

□ 中放入下述指令。

FOR
NEXT

输入自变量， EN: 执行条件 : 位
n: FOR ～ NEXT 之间的循环次数 (1 ～ 32767) : ANY16
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
n	○								-

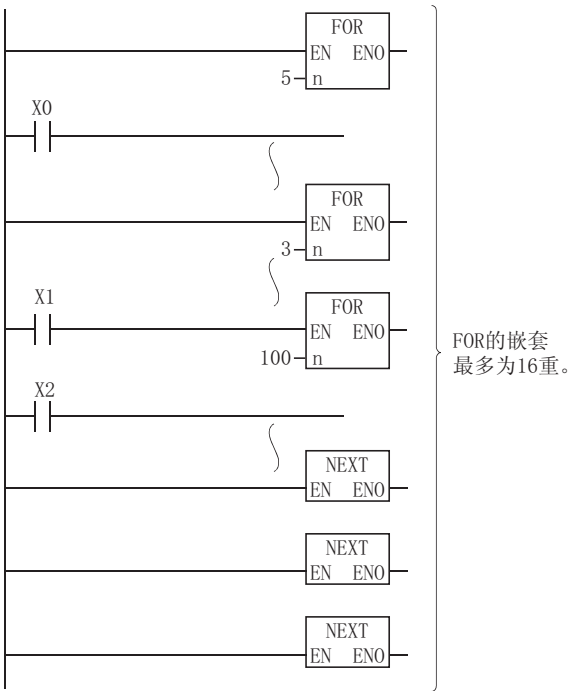
功 能

(1) 将 FOR ～ NEXT 指令之间的处理无条件地执行 n 次后，执行 NEXT 指令的下一步的处理。

(2) n 中可指定的范围为 1 ～ 32767。指定为 -32768 ～ 0 时，执行与 n=1 相同的处理。

(3) 不希望执行 FOR ～ NEXT 指令之间的处理时，应通过 CJ、SCJ 指令进行跳转。

(4) FOR 的嵌套最多为 16 重。



出 错

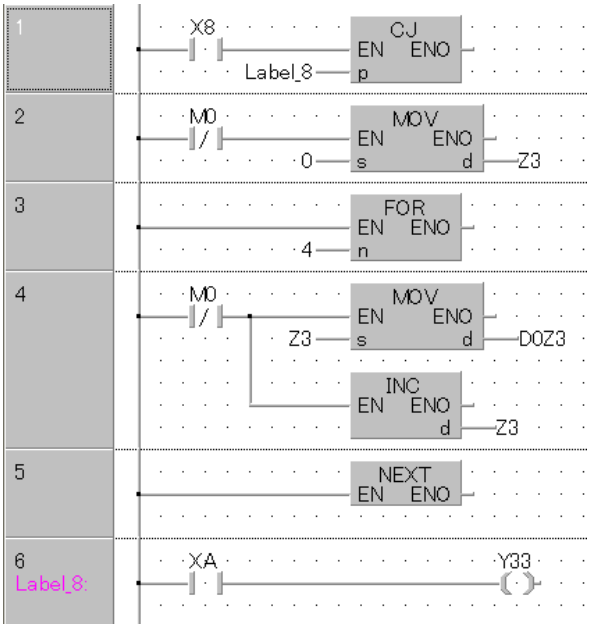
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 FOR 指令后，执行 NEXT 指令之前执行了 FEND、GOEND 指令时。 (出错代码： 4200)
- 执行 FOR 指令之前执行了 NEXT 指令时。 (出错代码： 4201)
- 在 FOR ～ NEXT 之间存在有 STOP 指令时。 (出错代码： 4200)
- 进行了 FOR 指令的嵌套的情况下，执行了第 17 重时。 (出错代码： 4202)

程序示例

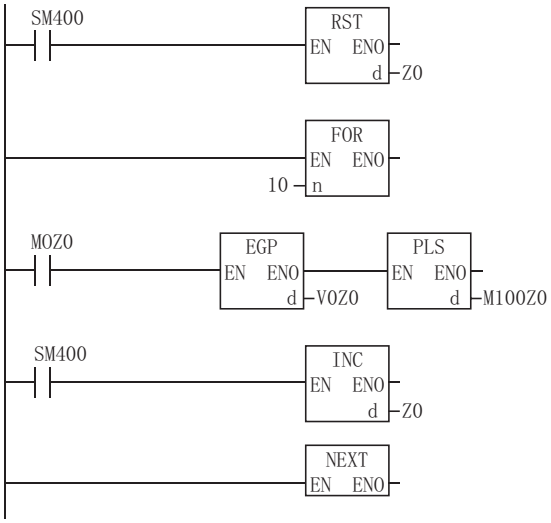
以下为 X8 为 OFF 时执行 FOR ～ NEXT 指令，X8 为 ON 时不执行 FOR ～ NEXT 指令的程序。

［结构体梯形图］



备 注

1. 在循环执行 FOR ～ NEXT 之间的过程中希望使其结束时，应使用 BREAK 指令。
   关于 BREAK 指令的详细内容，请参阅 7.6.2 项。
2. 在 FOR ～ NEXT 之间执行已变址修饰的程序的脉冲运算的情况下，应使用 EGP/EGF 指令。
   关于 EGP/EGF 指令的详细内容，请参阅 5.2.5 项。其样本程序如下所示。



3. 不能从 FOR ～ NEXT 的外面通过 JMP 指令等分支到 FOR ～ NEXT 以内。

7.6.2 FOR ～ NEXT 强制结束

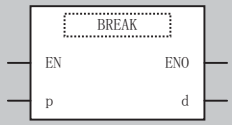
BREAK

Basic | High performance | Universal | L CPU

BREAK(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

中放入下述指令。

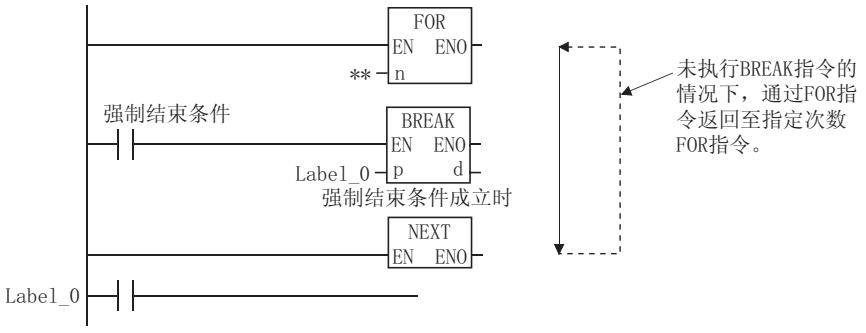
BREAK　　BREAKP

输入自变量，EN：　执行条件　：位
　　p：　使循环处理强制结束时的分支目标指针编号　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储循环剩余数的软元件编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				P
p	－							－	○
ⓓ	○							－	－

功 能

(1) 通过 FOR ～ NEXT 指令强制结束循环处理后，将执行转移至 Pn 中指定的指针处。
Pn 中只能指定同一程序文件内的指针。
Pn 中指定了其它程序文件内的指针的情况下，将变为运算出错状态。



(2) ⓓ 中存储强制结束时的 FOR ～ NEXT 指令中循环处理执行次数的剩余数
但是，循环处理的剩余数中还包含执行 BREAK(P) 指令时的次数。

(3) BREAK(P) 指令只能在 FOR ～ NEXT 指令之间使用。

(4) BREAK(P) 指令只能对 1 个嵌套使用。
对多重嵌套执行强制结束的情况下，应执行与嵌套重数对应次数的 BREAK(P) 指令。

出 错

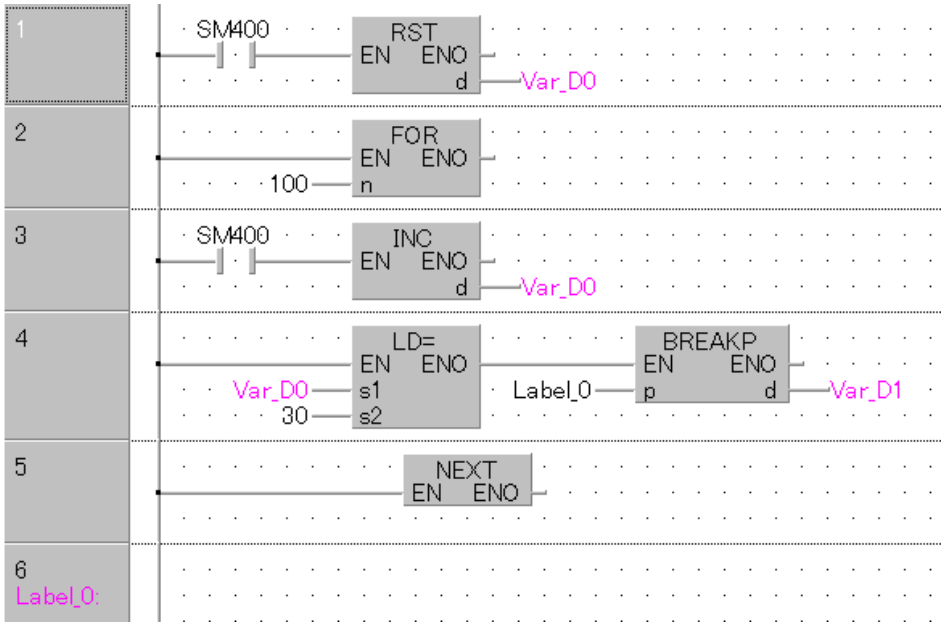
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 在 FOR ～ NEXT 指令以外使用了 BREAK(P) 指令时。 (出错代码： 4203)
- Pn 中指定的指针的跳转目标不存在时。 (出错代码： 4210)
- Pn 中指定了其它程序文件的指针时。 (出错代码： 4210)

程序示例

以下为 Var_D0 变为 30 时 (执行了 30 次的 FOR ～ NEXT 时)，对 FOR ～ NEXT 之间进行强制结束的程序。

[结构体梯形图]



备 注

执行 BREAK(P) 指令时，在 Var_D1 中将存储 71。

7.6.3 子程序调用

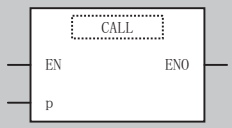
CALL

Basic | High performance | Universal | L CPU

CALL(P)

P：执行条件　　：

结构体梯形图



ST

不支持。

□ 中放入下述指令。

CALL　　CALLP

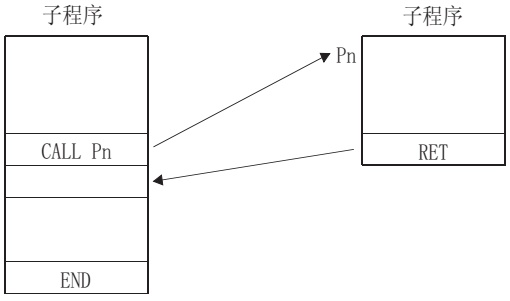
输入自变量，EN：　执行条件　：位
　　p：　子程序的起始指针编号　：ANY16
输出自变量，ENO：　执行结果　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它 P
	位	字		位	字				
p	–	–		–					○

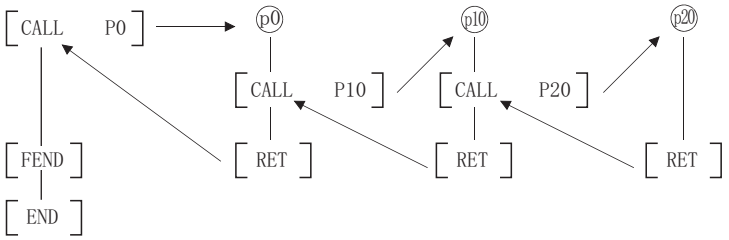
功 能

(1) 如果执行 CALL(P) 指令，则执行 Pn 中指定的指针的子程序。

通过 CALL(P) 指令，可以执行同一程序文件内的指针所指定的子程序及公共指针所指定的子程序。



(2) CALL(P) 指令的嵌套最多可为 16 重。



(3) 对于在子程序内置为 ON 的软元件，即使子程序变为非执行时也仍然被保持为 ON。

出 错

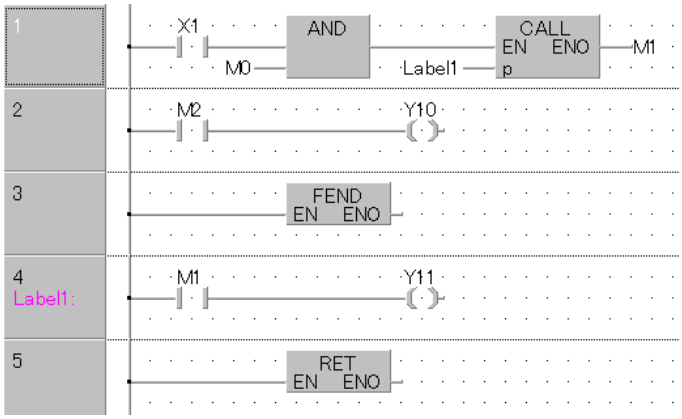
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 CALL(P) 指令后，在执行 RET 指令之前执行了 FEND、GOEND、STOP 指令时。 (出错代码： 4211)
- 在执行 CALL(P) 指令之前执行了 RET 指令时。 (出错代码： 4212)
- 执行了第 17 重的嵌套时。 (出错代码： 4213)
- CALL(P) 指令中指定指针的子程序不存在时。 (出错代码： 4210)

程序示例

以下为 X1 变为 ON 时，执行子程序的程序。

[结构体梯形图]



7.6.4 从子程序返回

RET

Basic | High performance | Universal | L CPU

RET

结构体梯形图

ST

ENO:= RET (EN);

□ 中放入下述指令。

RET

输入自变量， EN: 执行条件 :位

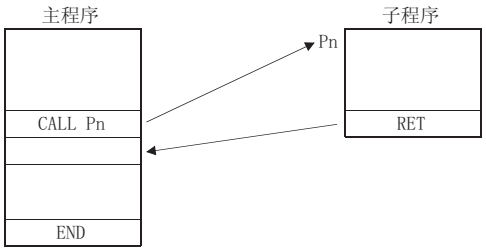
输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

(1) 表示子程序的结束。

(2) 如果执行 RET 指令，则返回至调用子程序的 CALL(P) 指令的下一步。



出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行 CALL(P) 指令后，在执行 RET 指令之前执行了 FEND、GOEND、STOP 指令时。 (出错代码： 4211)
- 在执行 CALL(P) 指令之前执行了 RET 指令时。 (出错代码： 4212)

7.6.5 刷新

COM

Basic High performance Universal L CPU

关于下述 CPU 模块的 COM 指令请参阅 7.6.6 项。

- 序列号的前 5 位数为 “04122” 以后的基本型 QCPU
- 序列号的前 5 位数为 “04012” 以后的高性能型 QCPU
- 通用型 QCPU
- LCPU

COM

结构体梯形图

ST

ENO:= COM (EN);

□ 中放入下述指令。

COM

输入自变量， EN: 执行条件 :位

输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

☆功 能

(1) COM 指令用于以下情况。

(a) 希望加快与远程 I/O 站的发送接收处理的情况下

(b) 数据链接执行过程中，希望与扫描时间不同的其它站进行数据收发的情况下

(2) COM 指令根据特殊继电器 SM775 的 ON/OFF 的不同其处理也有所不同。

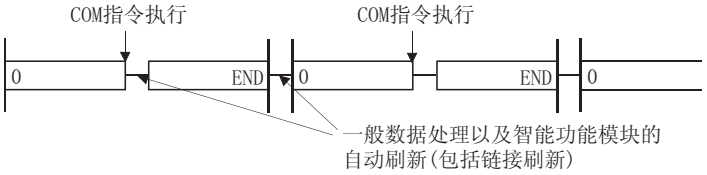
• SM775 为 OFF 时 : 进行自动刷新及与外围设备的通信。 *1*2

• SM775 为 ON 时 : 仅进行与外围设备的通信。 *1

*1 : 与外围设备的通信中还执行以下处理。
- 其它站监视处理
- 在串行通信模块中，对其它智能功能模块的缓冲存储器的读取处理

*2 : 在自动刷新处理中包含以下处理。
- MELSECNET/10、MELSECNET/H 的刷新
- CC-Link 的刷新
- 智能功能模块的自动刷新

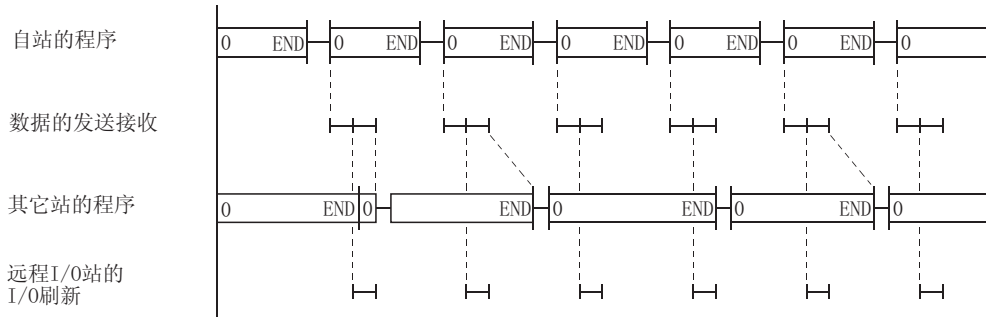
(3) 在执行 COM 指令时 CPU 模块暂时中断顺控程序的处理。此外，执行与 END 处理时的一般数据处理以及智能功能模块的自动刷新（包括链接刷新）相同的处理。
但是，不执行 MELSECNET/10、MELSECNET/H 的低速循环的刷新。



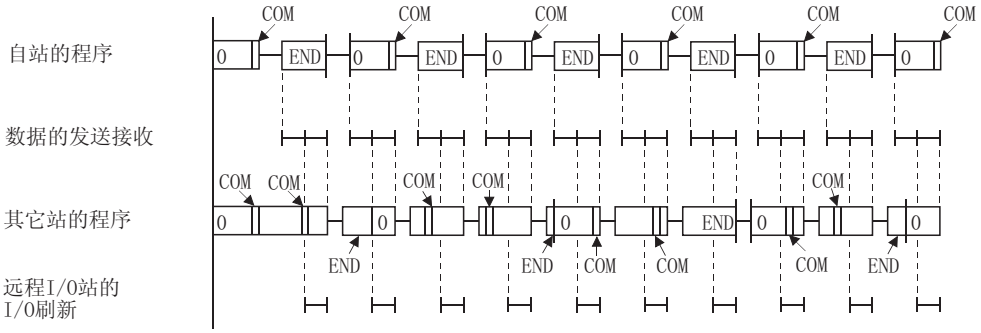
(4) COM 指令在顺控程序中的使用次数无限制。
但是，与外围设备的通信、智能功能模块的自动刷新（包括链接刷新）需要耗费时间，顺控程序的扫描时间将相应延长，应加以注意。

(5) 使用 COM 指令时的数据发送接收

(a) 未使用 COM 指令时的数据发送接收示例



(b) 使用了 COM 指令时的数据发送接收示例

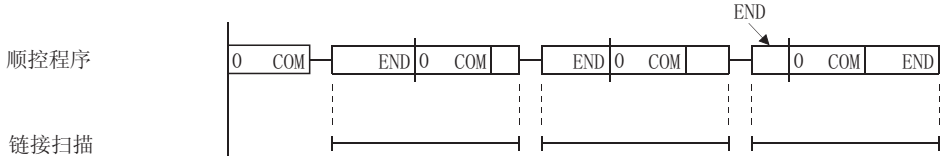


1) 如果对自站使用 COM 指令，如 (b) 项所示可以无条件地增加与远程 I/O 站的数据发送接收次数，可以提高数据的发送接收速度。
2) 其它站的扫描时间长于自站的扫描时间的情况下，通过对其它站侧使用 COM 指令，可以防止发生如 (a) 项所示的不能获取数据的时机。
3) 如果对其它站使用 COM 指令，则在下述期间接收到来自于自站的指令时各进行一次链接刷新。

- 第0步～COM指令
- COM指令～COM指令
- COM指令～END

各期间可各执行一次链接刷新。

(6) 自站的顺控程序的扫描时间短于链接扫描的时间时，即使对自站指定 COM 指令也不能提高数据的发送接收速度。



要 点

不能使用 COM 指令的程序如下所示。

- 低速执行类型程序
- 中断程序
- 恒定周期执行类型程序

出 错

不存在 COM 指令相关的运算出错。

7.6.6 选择刷新

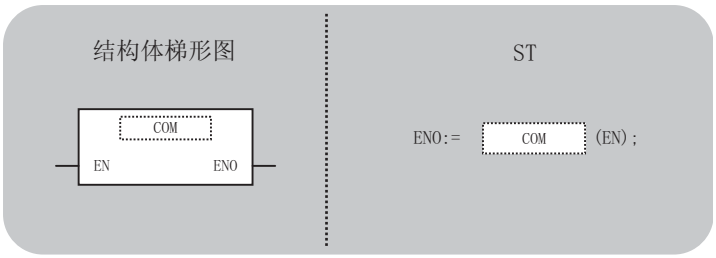
COM

关于下述 CPU 模块的 COM 指令请参阅 7.6.5 项。
• 序列号的前 5 位数为 “04121” 以前的基本型 QCPU
• 序列号的前 5 位数为 “04011” 以前的高性能型 QCPU

Ver. Basic Ver. High performance Universal L CPU

基本型 QCPU：序列号的前 5 位数为 “04122” 以后
高性能型 QCPU：序列号的前 5 位数为 “04012” 以后

COM



□ 中放入下述指令。

COM

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

(1) 如果执行 COM 指令，则可进行下述刷新。

刷新项目	QCPU(Q 模式)	LCPU
I/O 刷新	○	○
CC-Link 的刷新	○	○
CC-Link IE 控制网络的刷新	○	×
MELSECNET/H 的刷新	○	×
智能功能模块的自动刷新	○	○
使用了多 CPU 系统的 QCPU 标准区域的自动刷新	○	×
多 CPU 系统的组外的输入 / 输出的获取	○	×
使用了多 CPU 系统的多 CPU 间高速通信区域的自动刷新	○	×
与显示模块的通信	×	○
服务处理（与编程工具、GOT 或其它外部设备的通信）	○	○

备 注

在与外围设备的通信中还执行以下处理。
- 其它站监视处理
- 串行通信模块中，对其它智能功能模块的缓冲存储器的读取处理

(2) 如果将 SM775 置为 OFF，则对除 I/O 刷新以外的所有刷新项目进行刷新。

(3) 选择刷新项目时

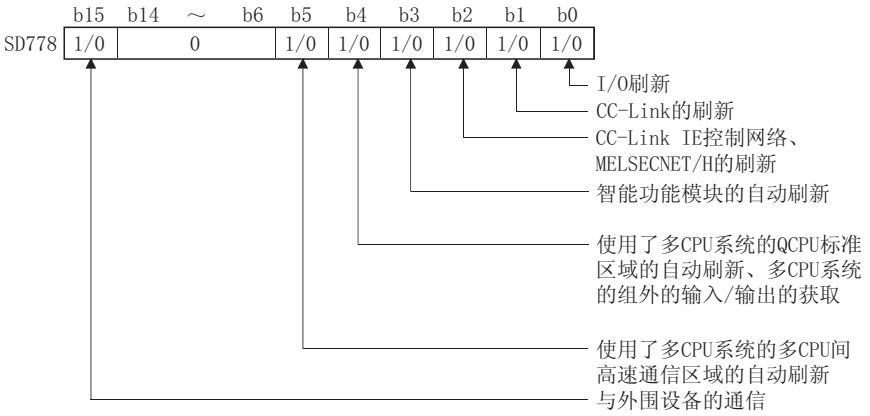
(a) 通过 SD778 选择刷新项目，将 SM775 置为 ON。
SM775 的 ON/OFF、SD778 中可指定的刷新项目如下表所示。

刷新项目	QCPU(Q 模式)		LCPU	
	SM775 为 OFF 时	SM775 为 ON 时	SM775 为 OFF 时	SM775 为 ON 时
I/O 刷新	非执行	可以对执行 / 非执行进行选择	非执行	可以对执行 / 非执行进行选择
CC-Link 的刷新	执行		执行	
CC-Link IE 控制网络的刷新			-	-
MELSECNET/H 的刷新			-	-
智能功能模块的自动刷新			执行	可以对执行 / 非执行进行选择
使用了多 CPU 系统的 QCPU 标准区域的自动刷新			-	-
多 CPU 系统的组外的输入 / 输出的获取			-	-
使用了多 CPU 系统的多 CPU 间高速通信区域的自动刷新			-	-
与显示模块的通信	-	-	执行	可以对执行 / 非执行进行选择
与外围设备的通信	执行	可以对执行 / 非执行进行选择		

(b) 通过 SD778 选择刷新项目。
SD778 的各个位的执行 / 非执行按下表所示进行指定。

1) QCPU(Q 模式) 的情况

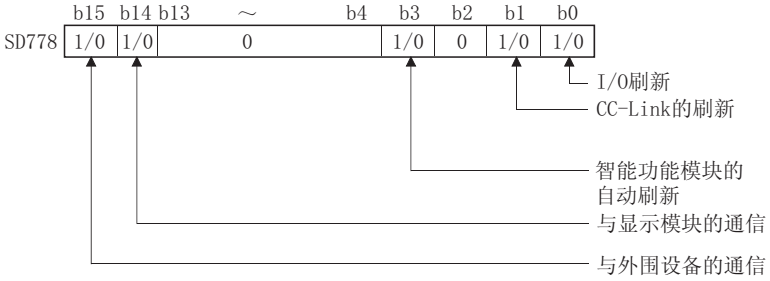
SD778 的位	执行	非执行
b0 ～ b5	1	0
b15	0	1



例 仅希望提高与远程 I/O 站的发送接收处理速度的情况下，仅指定 MELSECNET/H 的刷新。（仅在 SD778 的 b2 及 b15 中写入 1(SD778：8004H)）

2) LCPU 的情况

SD778 的位	执行	非执行
b0 ～ b3，b14	1	0
b15	0	1

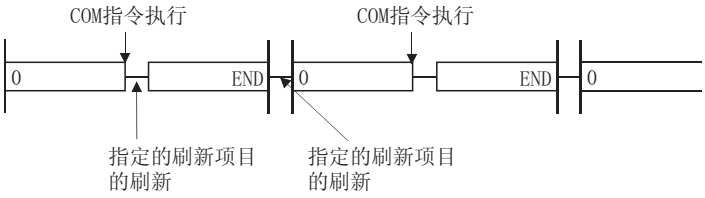


例 仅希望提高显示模块的处理速度的情况下，仅指定与显示模块的通信。（在 SD778 的 b14 及 b15 中写入 1(SD778：C000H)）

要 点

通过 COM 指令进行的多 CPU 间刷新在下述情况下执行。
- 来自于其它机号的接收动作： SD778 的 b4(CPU 共享存储器的自动刷新）为 1 时
- 来自于本站的发送动作： SD778 的 b15（与外围设备的通信的执行 / 非执行）为 0 时

(4) 在执行 COM 指令时 CPU 模块暂时中断顺控程序的处理，对指定的刷新项目进行刷新。



(5) COM 指令在顺控程序中的使用次数无限制。

但是，通过 SD778 选择的刷新项目需要耗费刷新时间，顺控程序的扫描时间将相应延长，应加以注意。

(6) 通用型 QCPU、LCPU 的情况下，的 COM 指令的执行过程中，将变为中断允许状态。但是，在中断程序等中使用刷新数据的情况下，有可能发生数据背离，应加以注意。

(7) 在以太网端口内置 QCPU、LCPU 中，在 CPU 内置以太网端口连接了以太网的状态下、通过 COM 指令执行了服务处理的情况下，处理时间将可能相应延长。

(8) COM 指令的刷新项目如下表所示。

CPU 模块型号	SM775	刷新项目
Q00J Q00 Q01	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	ON	仅与外围设备进行的通信
	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	ON	对通过 SD778 选择的刷新项目进行刷新
Q02 Q02H Q06H Q12H Q25H	ON/OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	ON	仅与外围设备进行的通信
	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	ON	对通过 SD778 选择的刷新项目进行刷新
Q00UJ Q00U Q01U Q02U Q03UD Q03UDE	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
Q04UDH Q04UDEH Q06UDH Q06UDEH Q10UDH Q10UDEH Q13UDH Q13UDEH Q20UDH Q20UDEH Q26UDH Q26UDEH	ON	对通过 SD778 选择的刷新项目进行刷新
L02 L26-BT	OFF	对除 I/O 刷新以外的刷新项目全部进行刷新
	ON	对通过 SD778 选择的刷新项目进行刷新

要 点

1. 在低速执行类型程序、恒定周期执行类型程序及中断程序中不能使用 COM 指令。

7.6.7 选择刷新

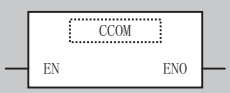
CCOM

Basic High performance Ver. Universal L CPU

• QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
• QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

CCOM(P)

结构体梯形图



ST

ENO:= CCOM (EN);

中放入下述指令。

CCOM　　CCOMP

输入自变量， EN:　　执行条件　　：位
输出自变量， ENO:　　执行结果　　：位

设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

关于功能，请参阅 7.6.6 项。

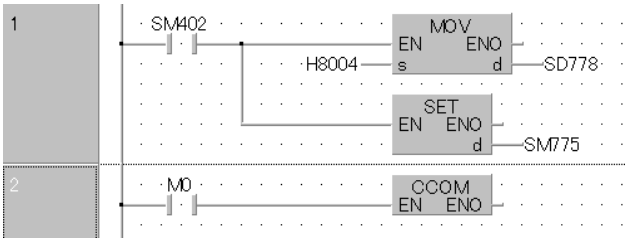
出 错

在序列号的前 5 位数为 “10101”以前的 QnU(D)(H)CPU 中执行了 CCOM(P) 指令时。
（出错代码： 4100）

程序示例

以下为根据 M0 的 ON/OFF，可对选择刷新的执行 / 非执行进行切换的程序。

［结构体梯形图］



7.7 数据表操作指令

7.7.1 至数据表的数据写入

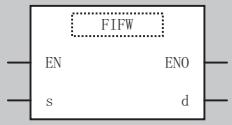
FIFW

Basic | High performance | Universal | L CPU

FIFW(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= FIFW (EN, s, d);
```

[ ] 中放入下述指令。

FIFW FIFWP

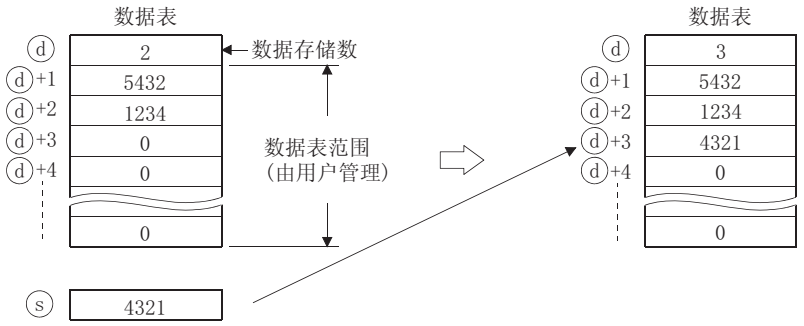
输入自变量， EN: 执行条件 : 位
s: 写入到表中的数据或存储数据的软元件编号 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 表的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○				○	-
ⓓ	-	○		-				-	-

功 能

(1) 将ⓢ中指定的 16 位数据存储到ⓓ中指定的数据表中。

在ⓓ中存储表中存储的数据数，ⓓ+1 以后依次存储ⓢ中指定的数据。



(2) 初次执行 FIFW(P) 指令的情况下，应将ⓓ中指定的软元件的值进行预先清除。

(3) 写入到表中的数据数及数据表范围应由用户进行管理。
（参阅程序示例 (2)）

出 错

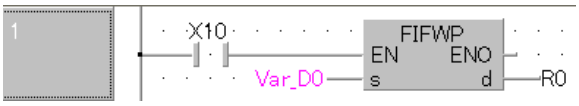
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行了 FIFW(P) 指令时，数据表范围超出了相应软元件的范围时。
（出错代码： 4101）

程序示例

(1) 以下为 X10 变为 ON 时，将 Var_D0 的数据存储到 R0 以后的数据表中的程序。

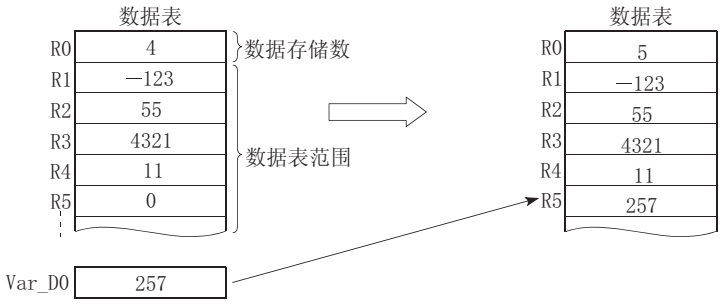
[结构体梯形图]



[ST]

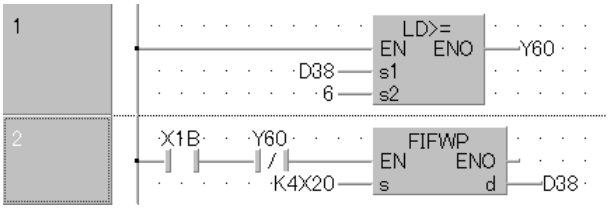
```
FIFWP(X10,Var_D0,R0);
```

[动作]



(2) 以下为 X1B 变为 ON 时，将 X20 ～ X2F 的数据存储到 D38 ～ D44 的数据表中，数据存储数超过了 6 的情况下，将 Y60 置为 ON 使 FIFW(P) 指令变为不能执行状态的程序。

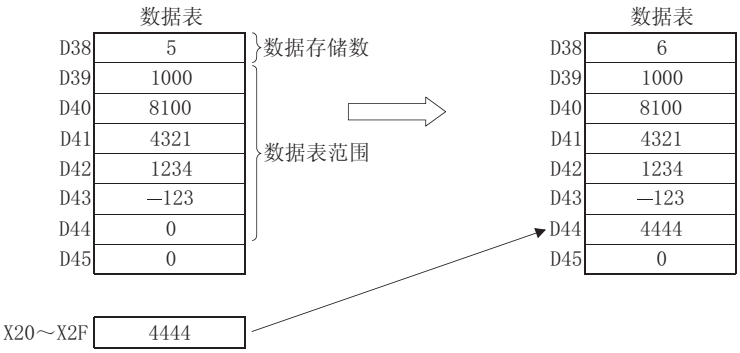
[结构体梯形图]



[ST]

```
OUT(D38>=6),Y60);
FIFWP(X1B AND NOT(Y60),K4X20,D38);
```

[动作]



7.7.2 数据表最前面数据的读取

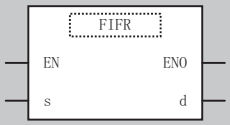
FIFR

Basic | High performance | Universal | L CPU

FIFR(P)

P: 执行条件　　: ↑

结构体梯形图



ST

ENO:= FIFR (EN, s, d);

[　　] 中放入下述指令。

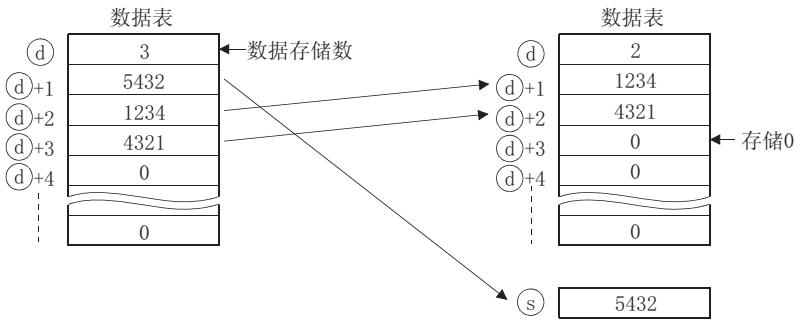
FIFR　　FIFRP

输入自变量，EN:　执行条件　：位
　　s:　存储从表中读取数据的软元件的起始编号　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　表的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	○	○		○				-	
ⓓ	-	○		-				-	

功 能

(1) 将ⓓ中指定的表的最前面的数据（ⓓ+1）存储到ⓢ中指定的软元件中。
执行 FIFR(P) 指令后数据表的数据逐个向前对齐。



(2) ⓓ中存储的值为 0 时，应由用户设置互锁使 FIFR(P) 指令变为不能执行状态。
（参阅程序示例 (1)）

出 错

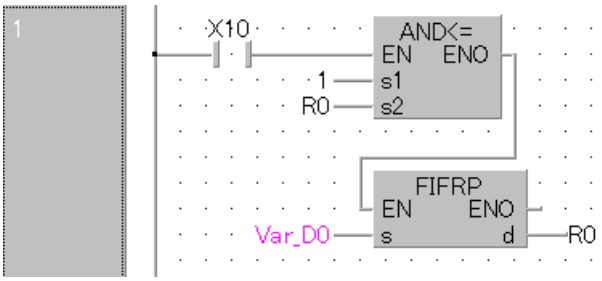
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 在ⓓ的值为 0 的状况下执行了 FIFR(P) 指令时。 （出错代码： 4100）
- 执行 FIFR(P) 指令时，数据表范围超出了相应软元件范围时。 （出错代码： 4101）

程序示例

(1) 以下为 X10 变为 ON 时，从 R0 ～ R7 的数据表中将 R1 的数据存储到 Var_D0 中的程序。

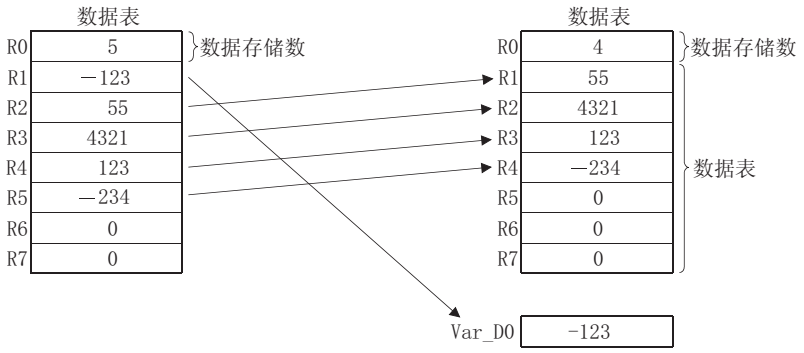
[结构体梯形图]



[ST]

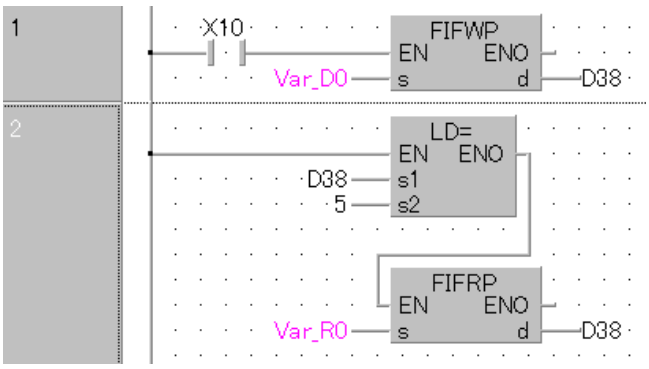
```
FIFRP(X10 AND R0>=1,Var_D0,R0);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将 Var_D0 的数据存储到 D38 ～ D43 的数据表中，数据存储数变为 5 时，将数据表的 D39 的数据存储到 Var_R0 中的程序。

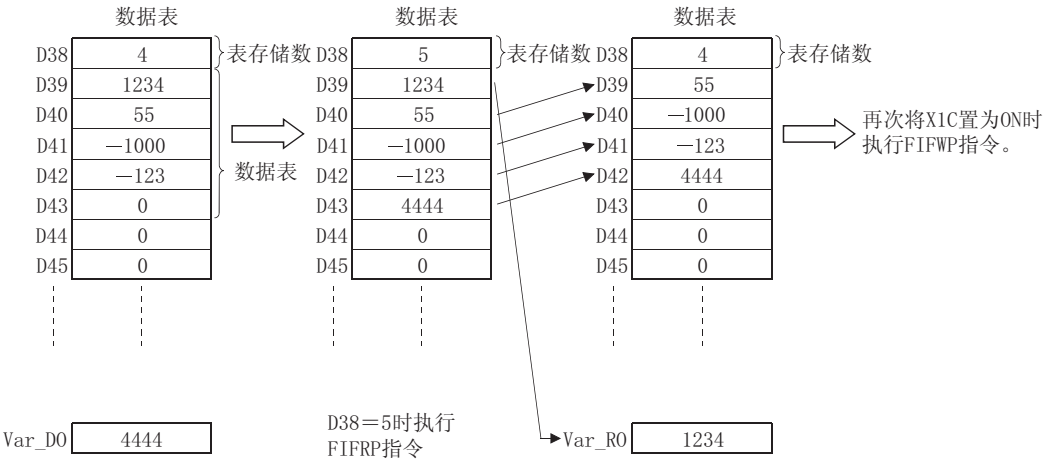
[结构体梯形图]



[ST]

```
FIFWP(X1C,Var_D0,D38);
FIFRP(D38=5,Var_R0,D38);
```

[动作]



7.7.3 数据表最后面数据的读取

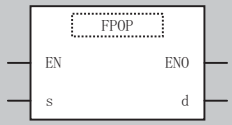
FP0P

Basic | High performance | Universal | L CPU

FP0P(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= FP0P (EN, s, d);

□ 中放入下述指令。

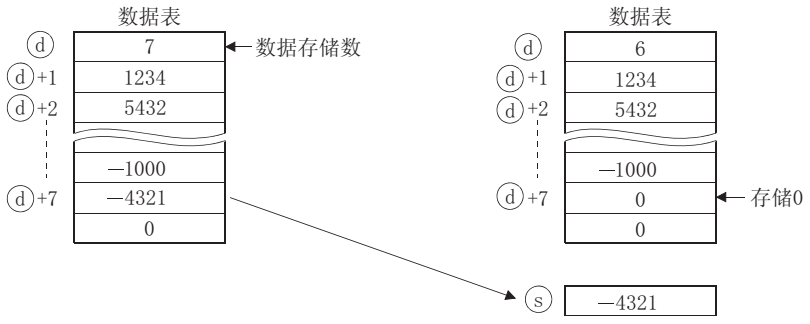
FP0P　　FP0PP

输入自变量，EN:　执行条件　：位
　　s:　存储从表中读取数据的软元件的起始编号　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　表的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	○	○		○				-	
ⓓ	-	○		-				-	

功 能

(1) 将存储在ⓓ中指定的表的最后面的数据存储到ⓢ中指定的软元件中。
执行 FP0P(P) 指令后，存储通过 FP0P(P) 指令读取的数据的软元件将变为 0。



(2) ⓓ中存储的值为 0 时，应由用户设置互锁使 FP0P(P) 指令指令变为不能执行状态。
(参阅程序示例 (1))

出 错

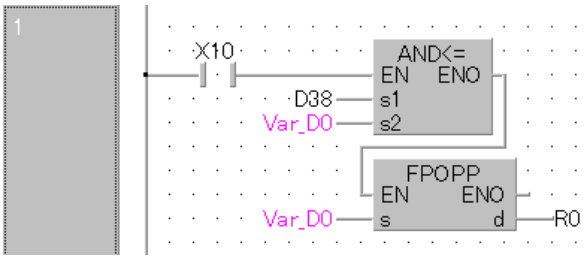
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 在ⓓ的值为 0 的状况下执行了 FPOP(P) 指令时。　　（出错代码： 4100）
- 执行 FPOP(P) 指令时，数据表范围超出了相应软元件范围时。　　（出错代码： 4101）

程序示例

(1) 以下为 X10 变为 ON 时，将存储在 R0 ～ R7 的数据表的最后的数据存储到 Var_D0 中的程序。

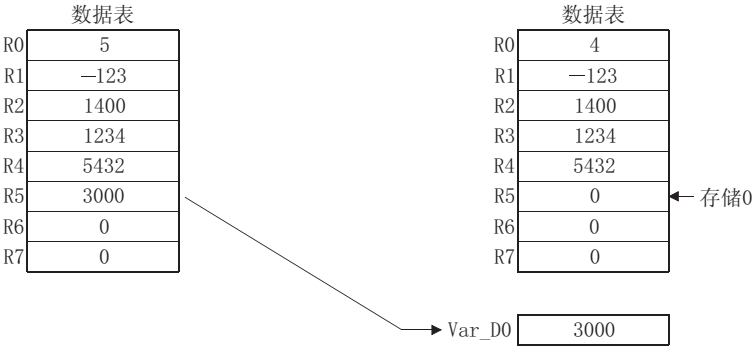
[结构体梯形图]



[ST]

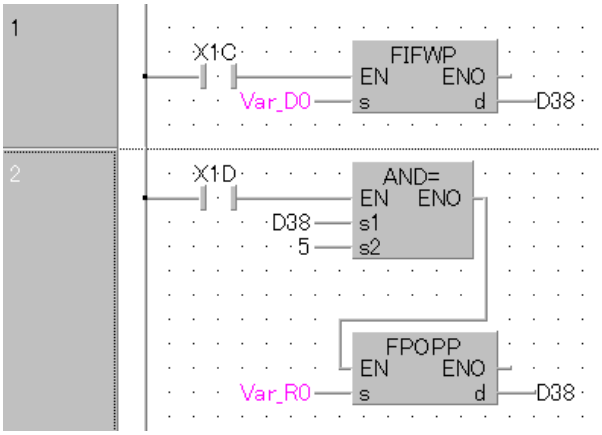
```
FPOPP(X10 AND Var_D0>=1,Var_D0,R0);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将 Var_D0 的数据存储到 D38 ～ D43 的数据表中，数据存储数为 5 时如果将 X1D 置为 ON，将存储在数据表的最后的数据存储到 Var_R0 中的程序。

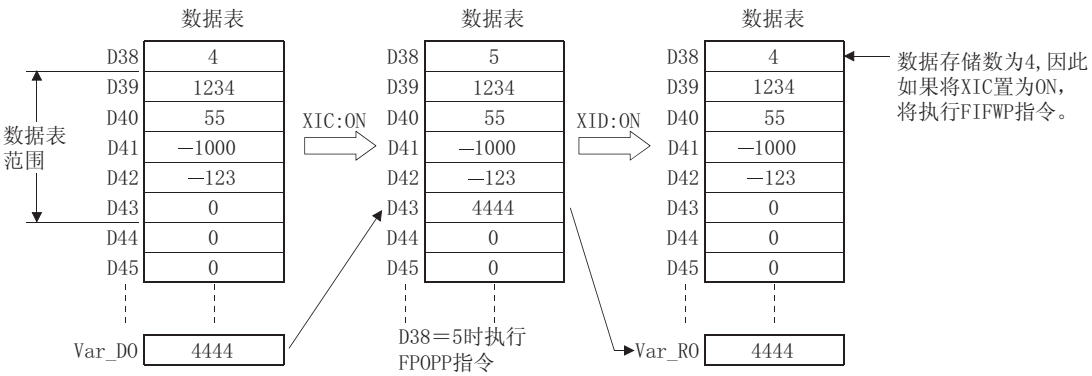
[结构体梯形图]



[ST]

```
FIFWP(X1C,Var_D0,D38);
FPOPP(X1D AND D38=5,Var_R0,D38);
```

[动作]



7.7.4 数据表的数据删除、插入

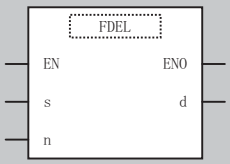
FDEL, FINS

Basic | High performance | Universal | L CPU

FDEL(P)
FINS(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= FDEL (EN, s, n, d);

□ 中放入下述指令。

FDEL	FDELP
FINS	FINSP

输入自变量， EN: 执行条件 : 位
s: 存储插入数据的软元件的起始编号 : ANY16
存储删除数据的软元件的起始编号
n: 插入 / 删除的表位置 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 表的起始编号 : ANY16

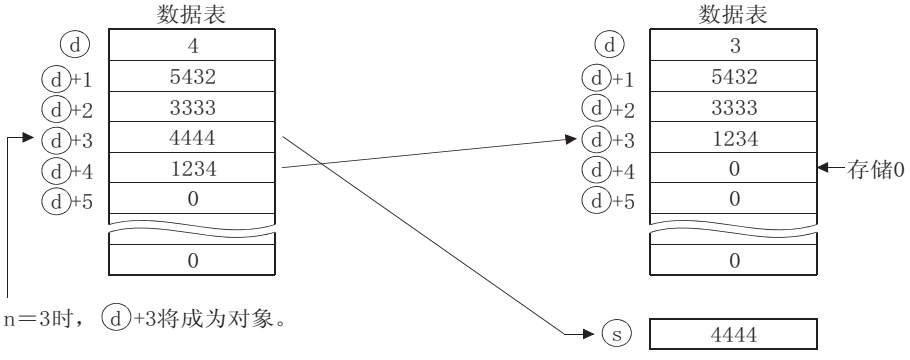
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○				–	–
n	○	○		○				○	–
ⓓ	–	○		–				–	–

功 能

FDEL(P)

将ⓓ中指定的数据表的第 n 号的数据删除后，存储到ⓢ中指定的软元件中。

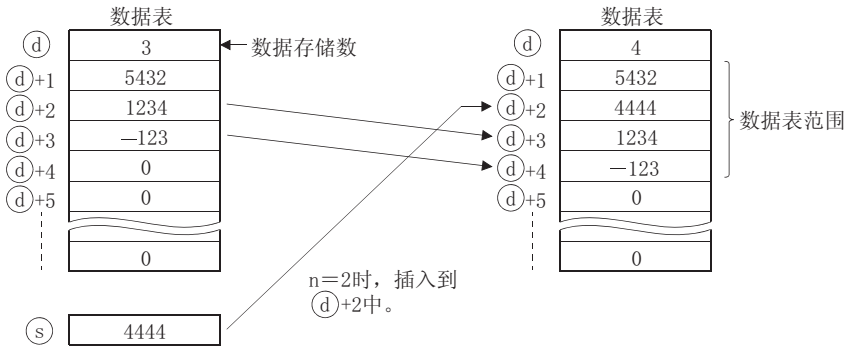
执行 FDEL(P) 指令后，数据表的第 n+1 号以后的数据将逐个向前对齐。



FINS(P)

将ⓢ中指定的 16 位数据，插入到ⓓ中指定的数据表的第 n 号中。

执行 FINS(P) 指令后，数据表的第 n 号以后的数据逐个下移。



出 错

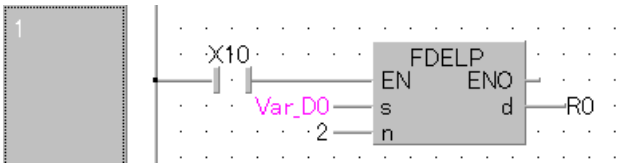
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- FDEL(P) 指令时，ⓓ算起第 n 号的位置大于数据存储数时。 (出错代码： 4101)
- FINS(P) 指令时，ⓓ算起第 n 号的位置大于数据存储数 +1 时。 (出错代码： 4101)
- FDEL(P),FINS(P) 指令时，n 的值超出了ⓓ的表的软元件范围时。 (出错代码： 4101)
- n=0 的情况下执行了 FDEL(P)、FINS(P) 指令时。 (出错代码： 4100)
- ⓓ的值为 0 的情况下执行了 FDEL(P) 指令时。 (出错代码： 4100)
- 执行了 FDEL(P)、FINS(P) 指令时，数据表范围超出了相应软元件范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X10 变为 ON 时，从 R0 ～ R7 的数据表中删除第 2 号的数据后，存储到 Var_D0 中的程序。

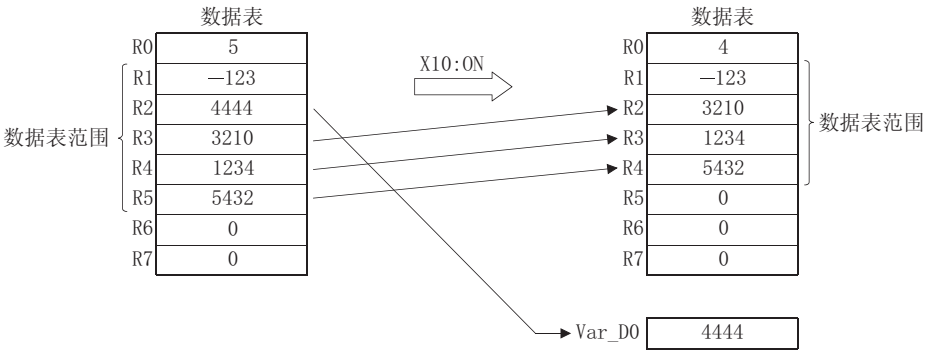
[结构体梯形图]



[ST]

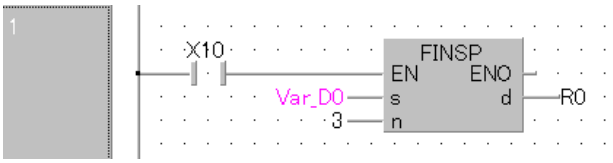
```
FDELP(X10,Var_D0,2,R0);
```

[动作]



(2) 以下为 X10 变为 ON 时，将 Var_D0 的数据插入到 R0 ～ R7 的表的第 3 号中的程序。

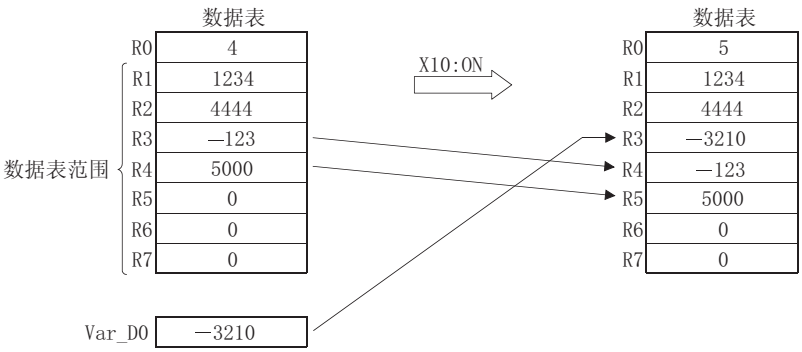
[结构体梯形图]



[ST]

```
FINSP(X10,Var_D0,3,R0);
```

[动作]



7.8 缓冲存储器访问指令

7.8.1 智能功能模块的 1 字、2 字数据读取

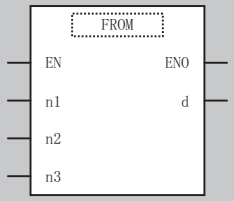
FROM, DFRO

Basic | High performance | Universal | L CPU

FROM(P)
DFRO(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= FROM (EN, n1, n2, n3, d);

□ 中放入下述指令。

FROM	FROMP
DFRO	DFROP

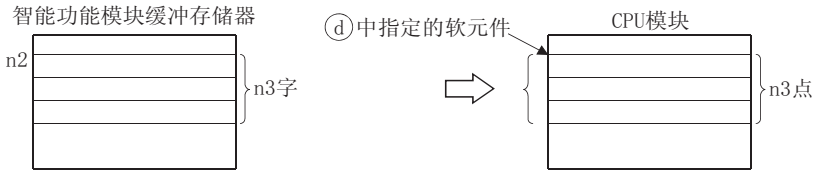
输入自变量，	EN:	执行条件	: 位
	n1:	智能功能模块的起始输入输出编号	: ANY16
	n2:	读取数据的起始地址	: ANY16
	n3:	读取数据数	: ANY16
输出自变量，	ENO:	执行结果	: 位
	d:	读取的数据	: ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它 U
	位	字		位	字				
n1	○			○				○	○
n2	○			○				○	-
n3	○			○				○	-
ⓓ	○			-				-	-

功 能

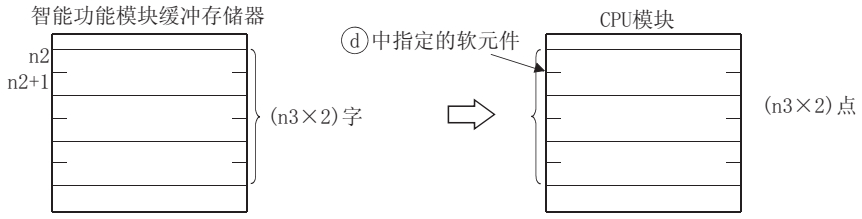
FROM(P)

从 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址中，读取 n3 字的数据后，存储到 ⓓ 中指定的软元件以后。



DFRO(P)

从 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址中，读取 (n3 × 2) 字的数据后，存储到ⓓ中指定的软元件以后。



要 点

智能功能模块的数据的读取也可使用智能功能模块软元件进行。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

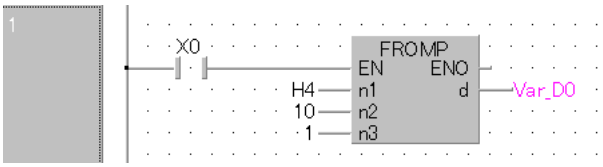
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行指令时不能与智能功能模块进行通信时。 (出错代码： 1412)
- 执行指令时检测出智能功能模块的异常时。 (出错代码： 1402)
- n1 中指定的输入输出编号不是智能功能模块时。 (出错代码： 2110)
- ⓓ中指定的软元件算起的 n3 点 (DFRO(P) 时为 2 × n3 点）超出了指定软元件范围时。 (出错代码： 4101)
- n2 中指定的地址超出了缓冲存储器的范围时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，在输入输出编号为 040 ～ 05F 的 A68AD 中，将 CH1 的数字值读取到 Var_D0 中的程序。（从缓冲存储器的地址 10 中读取 1 字的数据。）

[结构体梯形图]

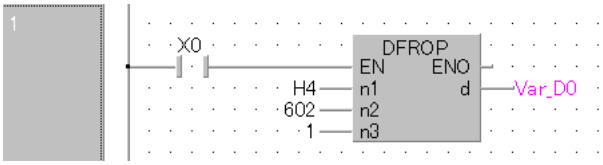


[ST]

```
FROMP(X0,H4,10,1,Var_D0);
```

(2) 以下为将 X0 置为 ON 时，在输入输出编号为 040 ～ 05F 的 AD71 中，将 X 轴的当前值读取到 Var_D0 中的程序。（从缓冲存储器的地址 602、603 中读取 2 字的数据。）

[结构体梯形图]



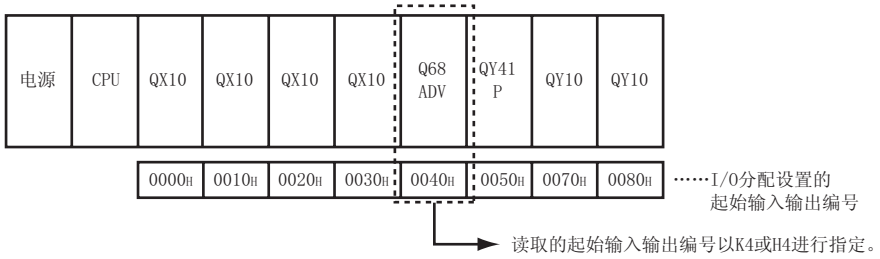
[ST]

```
DFROP(X0,H4,602,1,Var_D0);
```

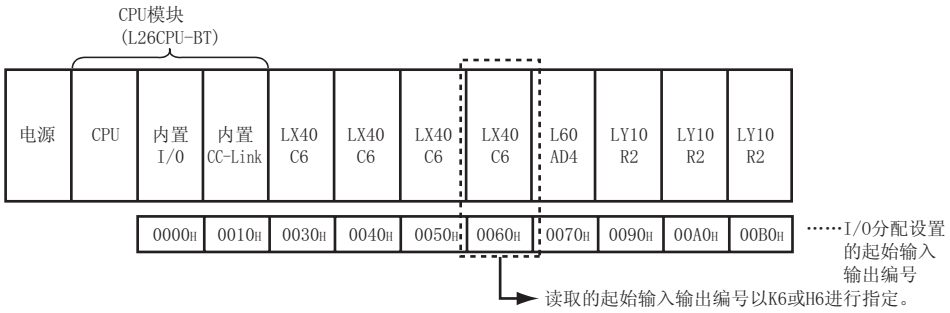
备 注

1. 将安装了智能功能模块的插槽的起始输入输出编号以 4 位的 16 进制数表示时的高 3 位指定到 n1 中。

〈QCPU(Q 模式)〉



〈LCPU〉



2. 在 QCPU(Q 模式)、LCPU 中，设置有 FROM/DFRO 指令的自动互锁。

7.8.2 至智能功能模块的 1 字、2 字数据写入

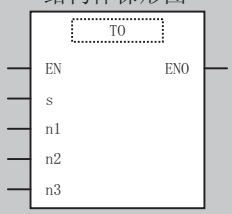
TO, DTO

Basic | High performance | Universal | L CPU

TO(P)
DTO(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= TO (EN, s, n1, n2, n3);

□ 中放入下述指令。

TO	TOP
DTO	DTOP

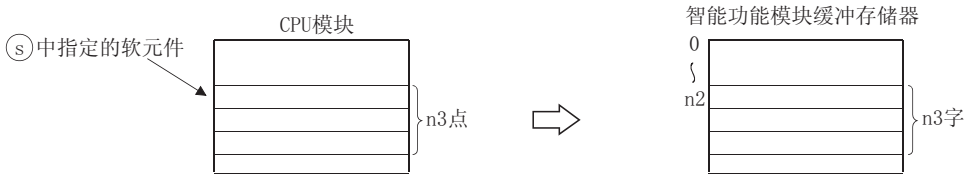
输入自变量，EN：　执行条件　：位
　　s：　写入数据　：ANY16/32
　　n1：　智能功能模块的起始输入输出编号　：ANY16
　　n2：　用于写入数据的起始软元件　：ANY16
　　n3：　写入数据数　：ANY16
输出自变量，ENO：　执行结果　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它 U
	位	字		位	字				
ⓢ		○			－			○	－
n1		○			○			○	○
n2		○			○			○	－
n3		○			○			○	－

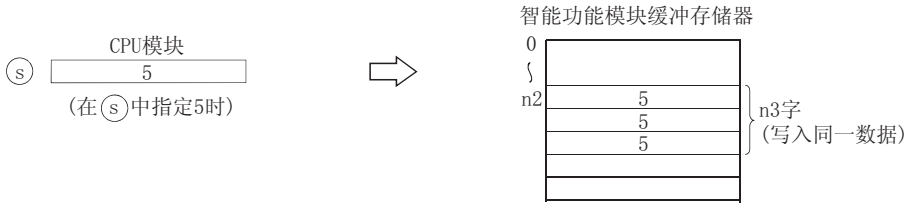
功 能

TO(P)

将ⓢ中指定的软元件算起 n3 点的数据，写入到 n1 中指定的智能功能模块的 n2 中指定的缓冲存储器地址以后。

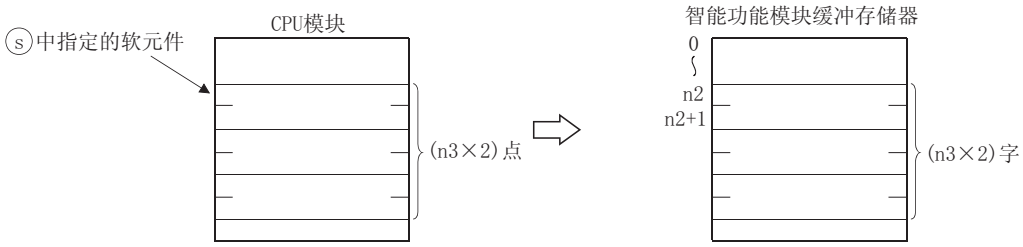


Ⓢ中指定了常数的情况下，将同一数据（Ⓢ中指定的值）写入到指定的缓冲存储器算起的 n3 字中。（Ⓢ中可指定的范围为 -32768 ～ 32767 或 0H ～ FFFFH。）

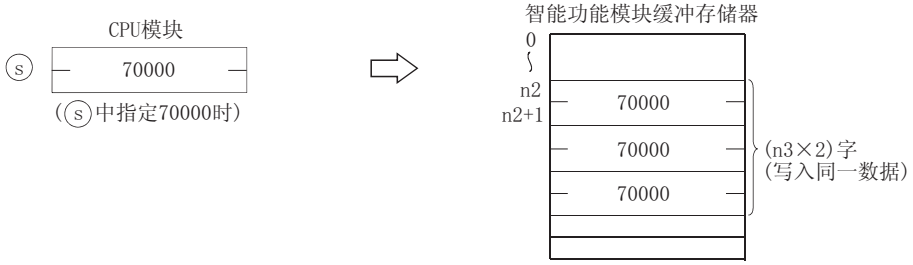


DTO(P)

将Ⓢ中指定的软元件算起 (n3 × 2) 点的数据，写入到 n1 中指定的智能機能模块的 n2 中指定的缓冲存储器地址以后。



Ⓢ中指定了常数的情况下，将同一数据（Ⓢ中指定的值）写入到指定的缓冲存储器算起的 (n3 × 2) 字中。
（Ⓢ中可指定的范围为 -2147483648 ～ 2147483647 或 0H ～ FFFFFFFFH。）



要 点

智能功能模块的数据的写入也可使用智能功能模块软元件进行。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

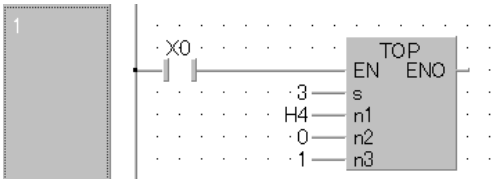
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 执行指令时不能与智能功能模块进行通信时。 （出错代码： 1412）
- 执行指令时检测出智能功能模块的异常时。 （出错代码： 1402）
- n1 中指定的输入输出编号不是智能功能模块时。 （出错代码： 2110）
- ⓢ中指定的软元件算起的 n3 点 (DTO(P) 时为 2 × n3 点) 超出了指定软元件范围时。 （出错代码： 4101）
- n2 中指定的地址超出了缓冲存储器的范围时。 （出错代码： 4101）
- n2 中指定的地址为奇数地址时。(AJ71QC24(N)) （出错代码： 4100）

程序示例

(1) 以下为 X0 变为 ON 时，在输入输出编号为 040 ～ 04F 的 Q68ADV 中，将 CH1 及 CH2 设置为“进行 A/D 转换”的程序。(在缓冲存储器的地址 0 中写入 3。)

[结构体梯形图]

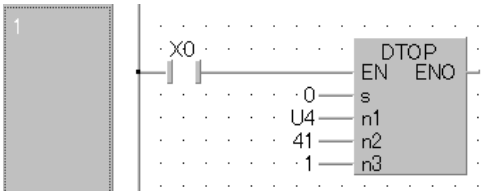


[ST]

```
TOP(X0,3,H4,0,1);
```

(2) 以下为将 X0 置为 ON 时，在输入输出编号为 040 ～ 05F 的 AD71 中，将 X 轴的当前值置为 0 的程序。(在缓冲存储器的地址 41、42 中写入 0。)

[结构体梯形图]



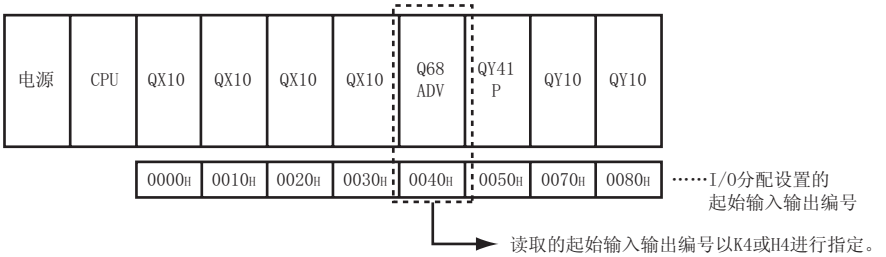
[ST]

```
DTOP(X0,0,U4,41,1);
```

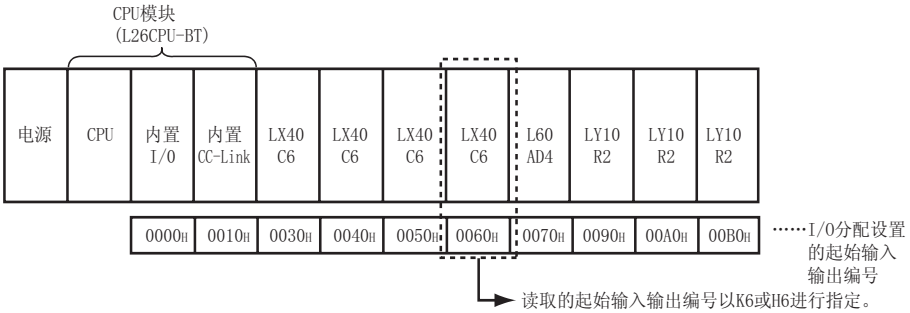
备 注

1. 将安装了智能功能模块的插槽的起始输入输出编号以 4 位的 16 进制数表示时的高 3 位指定到 n1 中。

〈QCPU(Q 模式)〉



〈LCPU〉



2. 在 QCPU(Q 模式)、LCPU 中设置有 TO/DTO 指令的自动互锁。

7.9 显示指令

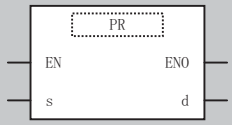
7.9.1 ASCII 码打印指令

PR

Basic | High performance | Universal | L CPU

PR

结构体梯形图



ST

```
ENO:=  PR  (EN, s, d);
```

□ 中放入下述指令。

PR

输入自变量， EN: 执行条件 : 位
s: ASCII 码 : ANY16/ 字符串
输出自变量， ENO: 执行结果 : 位
d: 输出 ASCII 码的输出模块的起始编号 : 位

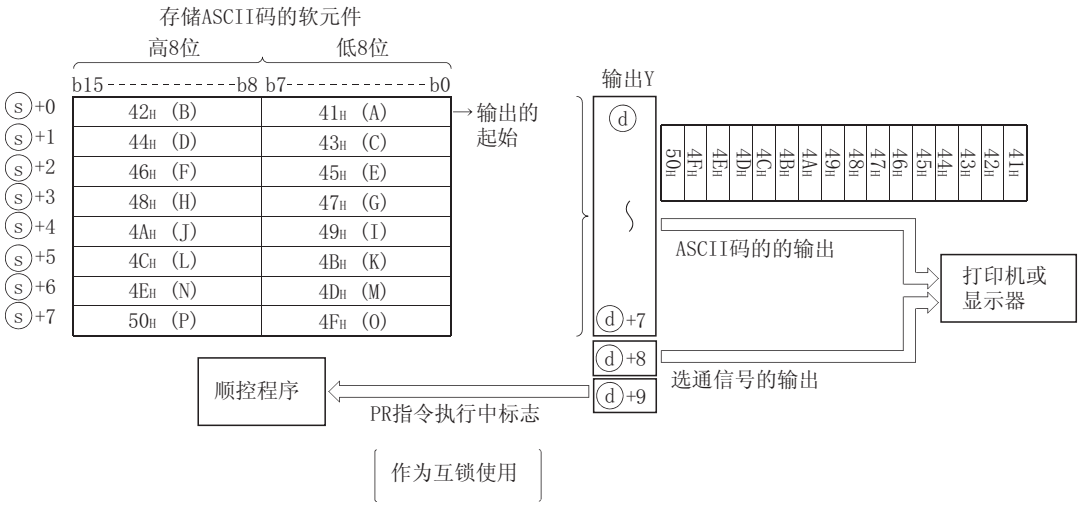
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	-	△*1		-			○	○	-
ⓓ	○(仅Y)	-		-			○	-	-

*1 ：不能使用局部软元件以及各程序中设置的文件寄存器。

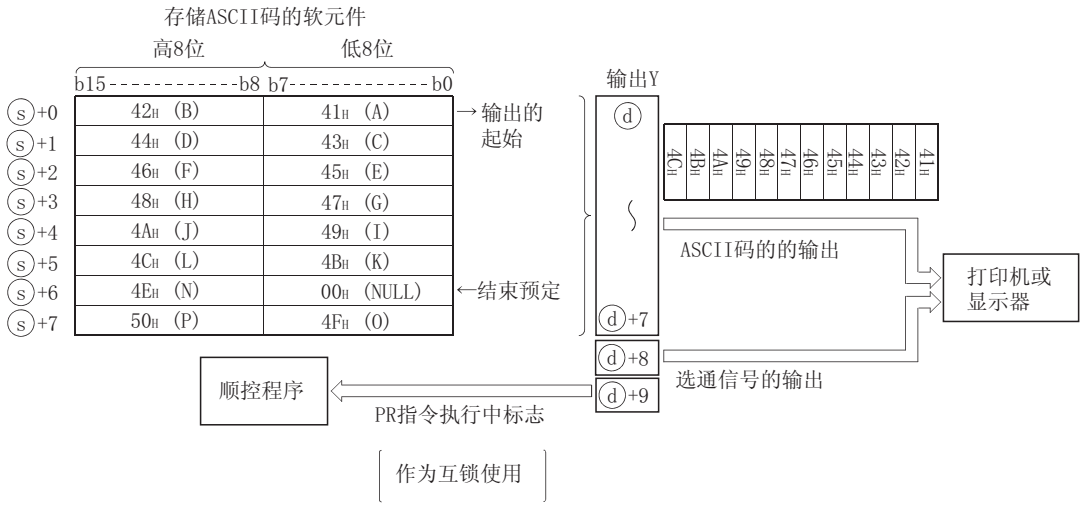
功 能

(1) 将ⓢ中指定的 ASCII 码或软元件编号以后存储的 ASCII 码输出到 ⓓ中指定的输出模块中。
输出的字符数根据 SM701(输出字符数切换)的 ON/OFF 而有所不同。

(a) SM701 为 ON 的情况下，ⓢ中指定的软元件算起的 8 点（16 字符）将成为对象。

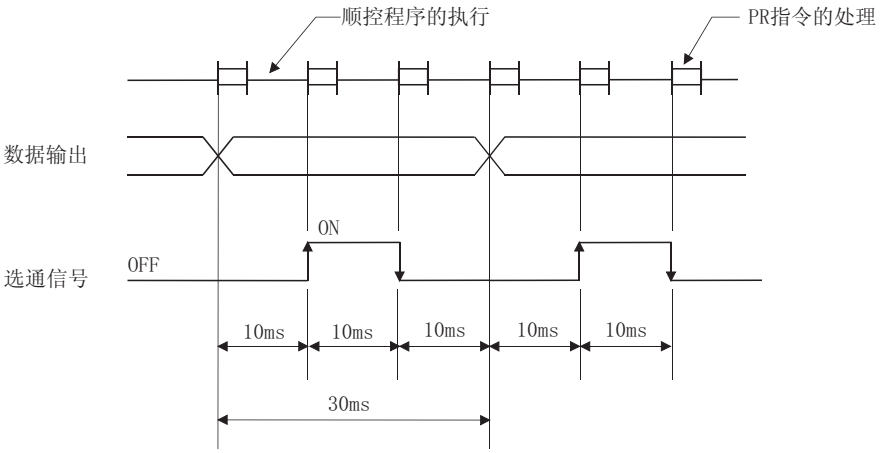


(b) SM701 为 OFF 的情况下，ⓢ中指定的软元件开始至 00H 码为止将成为对象。



(2) 输出模块中使用的点数为ⓓ中指定的 Y 编号算起的 10 点。

(3) 来自于输出模块的输出信号以 1 字符 30ms 进行发送。
因此，指定的字符数 (n) 的发送需要耗费 30ms × n(ms)。
PR 指令通过 10ms 中断，执行数据输出→选通信号 ON →选通信号 OFF。在执行上述处理与处理期间将继续执行其它指令。



(4) 除从输出模块输出 ASCII 码以外，还从ⓓ+8 的软元件输出选通信号 (10ms ON，20ms OFF)。

(5) 执行 PR 指令后，将 PR 指令执行标志（ⓓ+9 的软元件）置为 ON，直至指定字符的 ASCII 码的发送结束为止。

(6) 可以使用多个 PR、PRC 指令，但为了防止同时 ON，应通过 PR 指令执行中标志（ⓓ+9 的软元件）设置互锁。

(7) 对于存储 ASCII 码的软元件的内容，如果在 ASCII 码的输出过程中发生了变化，则将输出变更后的数据。

出 错

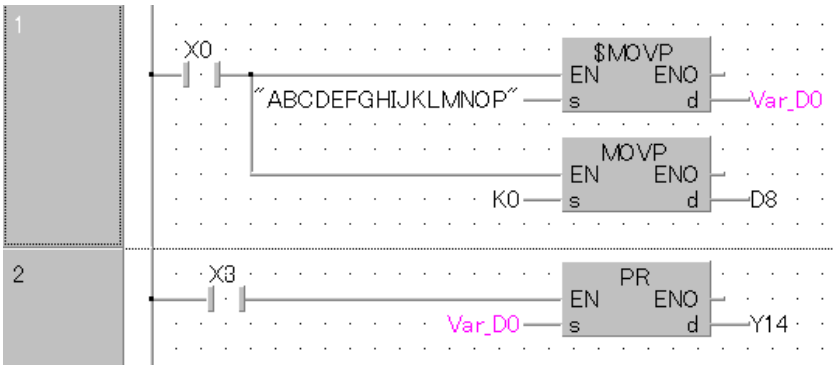
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- SM701 为 OFF 时，ⓢ中指定的软元件的范围内不存在 00_H 码时。　　（出错代码：4101）

程序示例

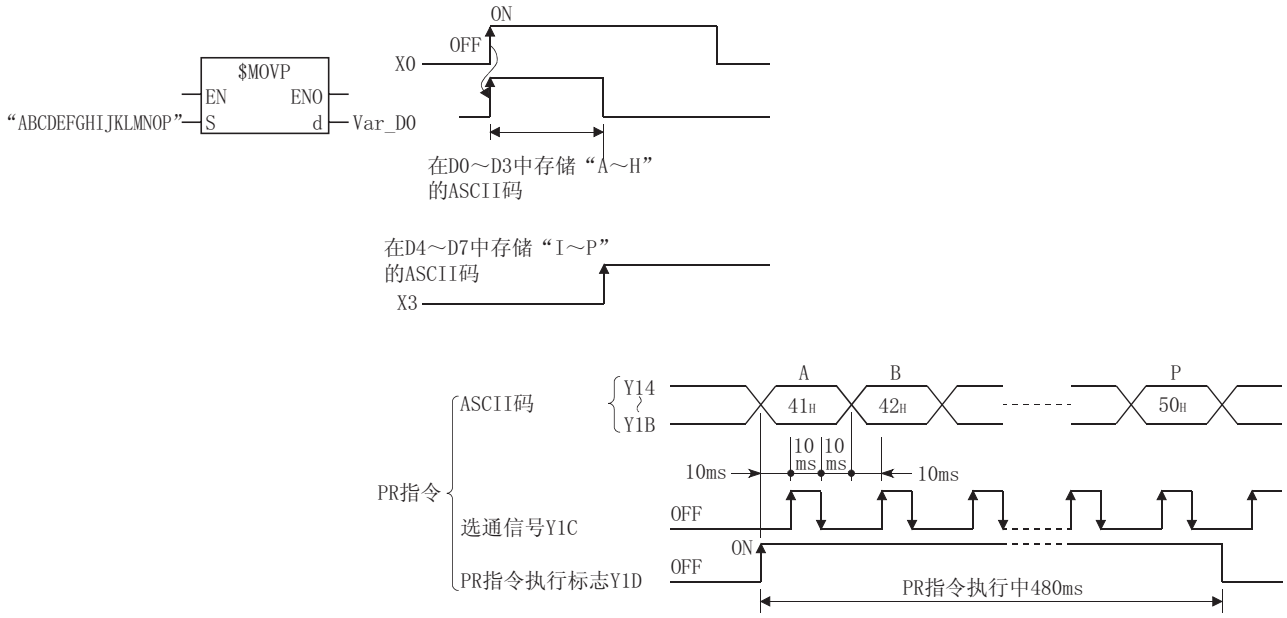
以下为 X0 变为 ON 时，将 “ABCDEFGHIJKLMNOP”转换为 ASCII 码后存储到 Var_D0 中，X3 变为 ON 时，将 Var_D0 的 ASCII 码输出到 Y14 ～ Y1D 的程序。

[结构体梯形图]




```
[ST]
IF X0 THEN
     Var_D0="ABCDEFGHIJKLMNOP";
     MOVP(TRUE,0,D8);
END_IF;
PR(X3,Var_D0,Y14);
```

[时序图]

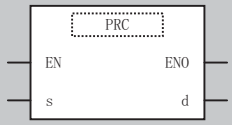


7.9.2 注释的打印指令

PRC

PRC

结构体梯形图



ST

ENO:= PRC (EN, s, d);

中放入下述指令。

PRC

输入自变量， EN: 执行条件 : 位
s: 打印注释的软元件的起始编号 : ANY_SIMPLE
输出自变量， ENO: 执行结果 : 位
d: 输出注释的输出模块的起始编号 : 位

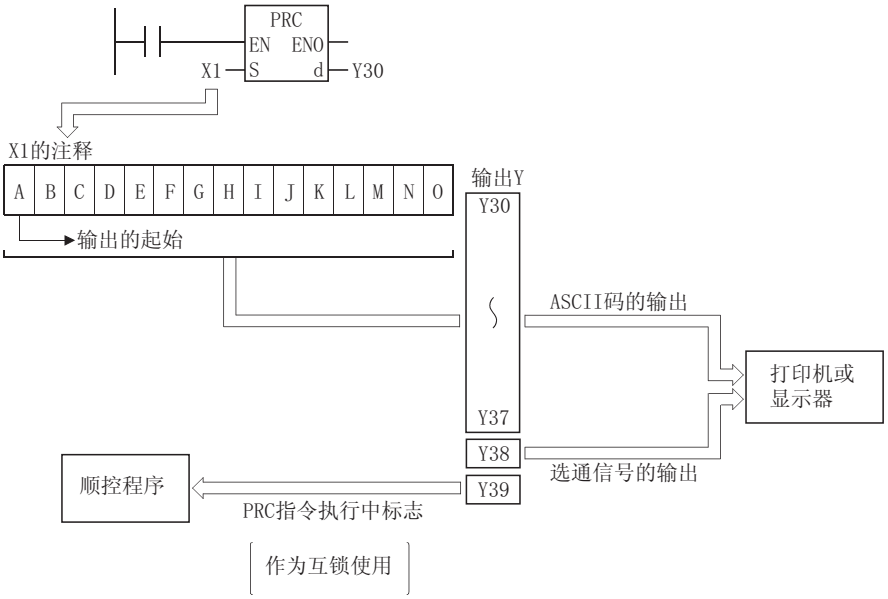
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它 P, I, J, U
	位	字		位	字				
ⓢ	○	○		○			-	-	○
ⓓ	○(仅Y)	-		-			-	-	-

功 能

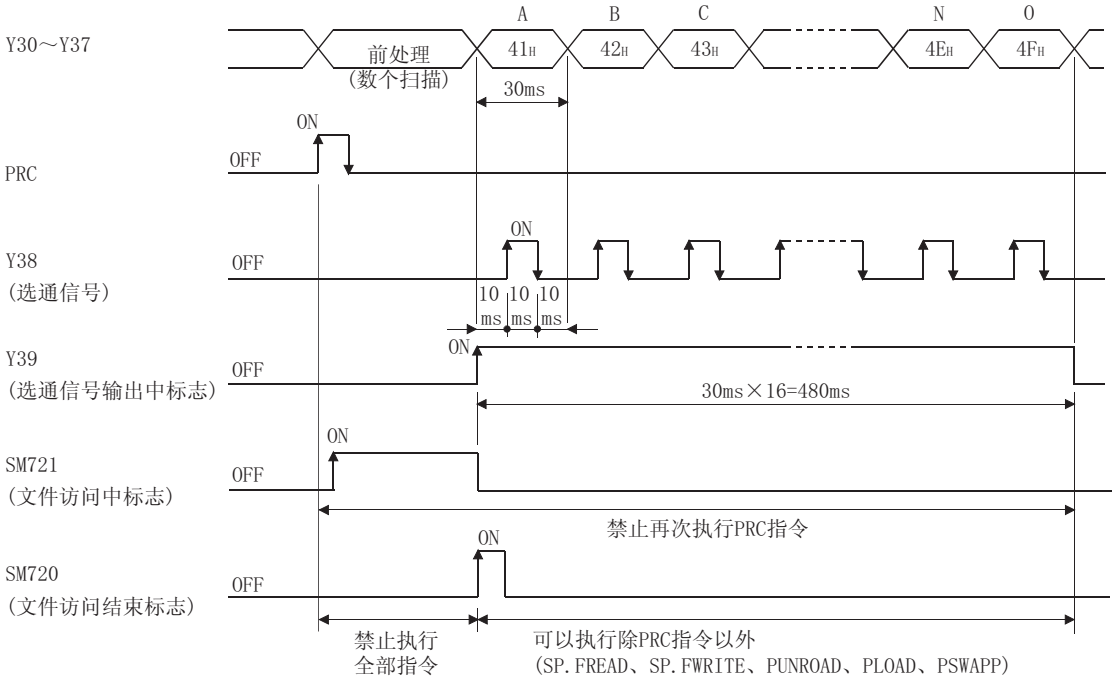
(1) 将ⓢ中指定的软元件的注释（ASCII 码）输出到ⓓ中指定的输出模块中。
输出的字符数根据 SM701 的 ON/OFF 而有所不同。

- SM701 为 OFF 时 ： 32 字符的注释
- SM701 为 ON 时 ： 高位 16 字符的注释

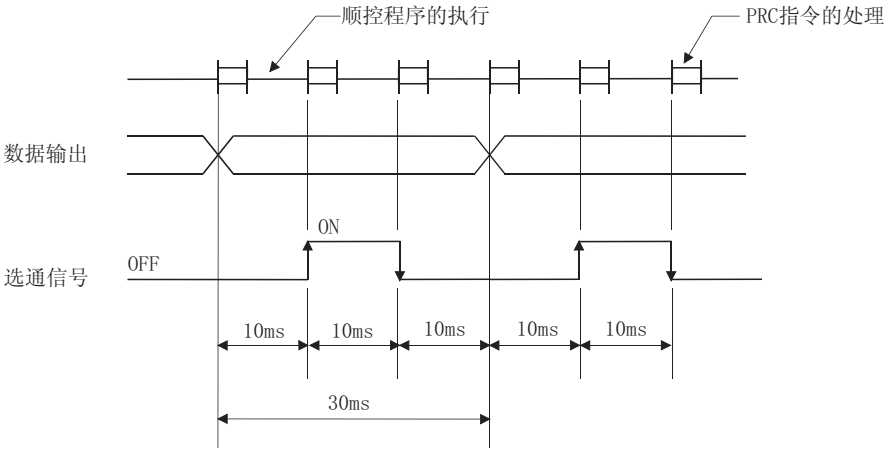
输出模块中使用的点数为ⓓ中指定的 Y 编号算起的 10 点。


PRC
EN ENO
X1 S d Y30
X1的注释
A B C D E F G H I J K L M N O
输出Y
Y30
Y37
输出的起始
ASCII码的输出
打印机或
显示器
Y38
选通信号的输出
顺控程序
Y39
PRC指令执行中标志
作为互锁使用

［时序图］


Y30～Y37
A
B
C
N
O
前处理
(数个扫描)
41H
42H
43H
4EH
4FH
30ms
PRC
ON
OFF
Y38
(选通信号)
10 ms
10 ms
10 ms
Y39
(选通信号输出中标志)
30ms×16=480ms
SM721
(文件访问中标志)
禁止再次执行PRC指令
SM720
(文件访问结束标志)
禁止执行
全部指令
可以执行除PRC指令以外
(SP.FREAD、SP.FWRITE、PUNROAD、PLOAD、PSWAPP)

(2) 来自于输出模块的输出信号以 1 字符 30ms 进行发送。
因此，指定的字符数 (n) 的发送需要耗费 30ms × n(ms)。
PRC 指令通过 10ms 中断，执行数据输出→选通信号 ON →选通信号 OFF。在执行上述处理与处理期间将继续执行其它指令。



(3) 除从输出模块输出 ASCII 码以外，还从ⓓ+8 的软元件输出选通信号 (10ms ON，20ms OFF)。

(4) 执行 PRC 指令后，将 PRC 指令执行标志（ⓓ+9 的软元件）置为 ON，直至指定字符的 ASCII 码的发送结束为止。

(5) 可以使用多个 PRC 指令，但为了防止同时 ON，应通过 PRC 指令执行中标志（ⓓ+9 的软元件）设置互锁。

(6) 注释未被登录到ⓢ中指定的软元件中的情况下将不执行处理。

(7) 读取注释时，指令结束后 SM720 将 1 个扫描 ON。
此外，指令执行过程中 SM721 将变为 ON。
在 SM721 为 ON 的状态下，不能执行 PRC 指令。如果执行则变为无处理。

要 点

1. PRC 指令中使用的软元件的注释使用存储卡中存储的注释文件。
   不能使用内置存储器中存储的注释文件。
2. PRC 指令中使用的注释文件是在参数模式的 “可编程控制器文件设置”中进行设置。
   未在可编程控制器文件设置中对使用的注释文件进行设置的情况下，将不能通过 PRC 指令输出软元件的注释。
3. 不要在中断程序中执行 PRC 指令。
   如果执行将导致误动作。

出 错

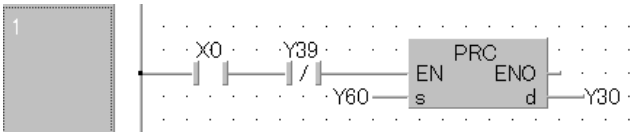
在以下情况下，将变为运算出错状态，出错标志将 ON。

- 在软元件注释的运行中写入过程中执行了 PRC 指令时。 （出错代码 :4100）

程序示例

以下为 X0 变为 ON 时，将 Y60 的注释输出到 Y30 ～ Y39 中的程序。

[结构体梯形图]



[ST]

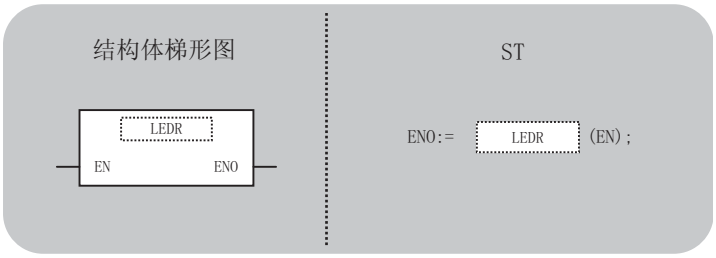
```
PRC(X0 AND NOT(Y39),Y60,Y30);
```

7.9.3 出错显示或报警器复位指令

LEDR

Basic High performance Universal L CPU

LEDR



中放入下述指令。

LEDR

输入自变量， EN: 执行条件 :位
输出自变量， ENO: 执行结果 :位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

功 能

对 CPU 模块的报警器显示及可继续运行的自诊断出错的显示进行复位。
在一次指令执行中，只能对出错显示或报警器这二者之一进行复位。

(1) 发生自诊断出错时的动作

(a) 发生了可继续运行的自诊断出错时。
在 CPU 模块显示为可继续运行的自诊断出错时，如果执行 LEDR 指令，将对 CPU 模块前面的 “ERROR/ERR.” LED 或出错显示进行复位。
此时由于 SM0、SM1、SD0 的内容不能被复位，因此应由用户程序进行复位。
此外，由于此时出错显示的出错原因的优先顺序高于报警器，因此不进行报警器复位的相关处理。

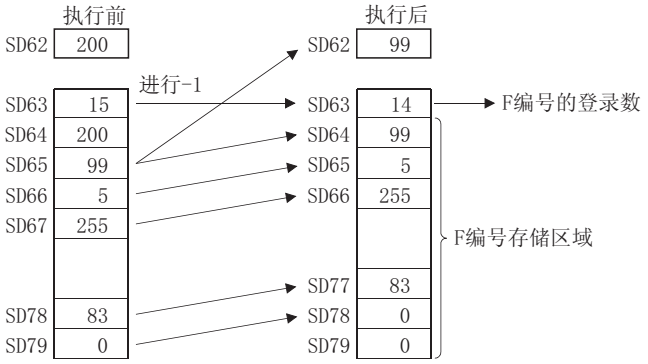
(b) 发生电池出错时
如果在替换电池后执行 LEDR 指令，CPU 模块前面的 “BAT.ARM/BAT.” LED 或出错显示将被复位。
此时 SM51 也将变为 OFF。

(2) 报警器 (F) 为 ON 时的动作

(a) 前面没有 LED 显示器的 CPU 模块的情况

执行 LEDR 指令时将进行以下动作。

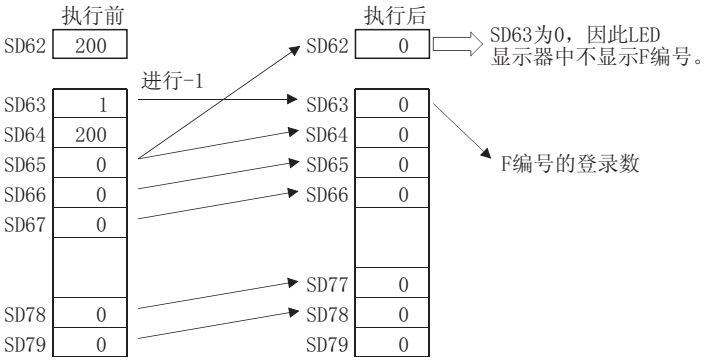
1) 进行 “USER” LED 的闪烁、熄灯。
2) 对 SD62、SD64 中存储的报警器 (F) 进行复位，SD65 ～ SD79 的 F 编号将向前对齐。
3) 将新存入 SD64 中的数据传送至 SD62 中。
4) 对 SD63 的数据进行 -1。但是 SD63 为 0 时保持为 0 不变。



(b) 前面有 LED 显示器的 CPU 模块的情况

执行 LEDR 指令时将进行以下动作。

1) 对 CPU 模块前面显示的 F 编号进行复位。
2) 进行 “USER” LED 的闪烁、熄灯。
3) 对 SD62、SD64 中存储的报警器 (F) 进行复位，SD65 ～ SD79 的 F 编号向前对齐。
4) 将新存入 SD64 中的数据传送至 SD62 中。
5) 对 SD63 的数据进行 -1。但是 SD63 为 0 时保持为 0 不变。
6) 将 SD62 中存储的 F 编号显示到 LED 显示器中。但是 SD63 为 0 时不显示任何内容。



备 注

1. 特殊寄存器 SD207 ～ SD209 中设置的原因编号的内容及默认的优先顺序如下所示。

优先顺序	原因编号（16 进制数）	内容	备注
1	1	AC DOWN SINGLE PS. DOWN SINGLE PS. ERROR	电源断 冗余基板的电源电压过低 冗余电源模块异常
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	输入输出模块校验出错 保险丝熔断 特殊功能模块校验出错
3	3	OPERATION ERROR LINK PARA. ERROR SFCP OPE. ERROR SFCP EXE. ERROR	运算出错 链接参数出错 SFC 指令运算出错 SFC 程序执行出错
4	4	ICM. OPE. ERROR FILE OPE. ERROR EXTEND INST. ERROR OPE. MODE DIFF. CAN'T EXE. MODE TRK.TRANS. ERR. TRK.SIZE ERROR TRK.DISCONNECT	存储卡操作出错 文件访问出错 扩展指令出错 运行状态、开关不一致 现有模式下不能执行此功能 热备数据通信出错 热备容量超过出错 热备电缆未连接・故障
5	5	PRG.TIME OVER	恒定扫描设置时间溢出 低速执行监视时间溢出
6	6	CHK 指令	—
7	7	报警器	—
8	8	LED 指令	—
9	9	BATTERY ERR.	—
10	A	时钟数据	—
11	B	CAN'T SWITCH STANDBY SYS. DOWN MEM. COPY EXE.	系统切换出错 待机系统未启动 / 停止出错 存储器复制功能实施

2. 如果将报警器设置为最优先，则可通过 LEDR 指令优先对报警器进行复位。

7.10 调试、故障诊断指令

7.10.1 特定格式故障检查

CHKST, CHK

Basic | High performance | Universal | L CPU

CHKST
CHK

结构体梯形图

ST

ENO:= CHKST (EN);

□ 中放入下述指令。

CHKST
CHK

输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
-	-								

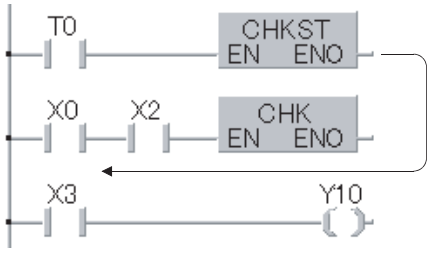
功 能

CHKST

CHKST 指令是表示 CHK 指令开始的指令。

CHKST 指令为 OFF 的情况下，跳转至 CHK 指令的下一个指令。

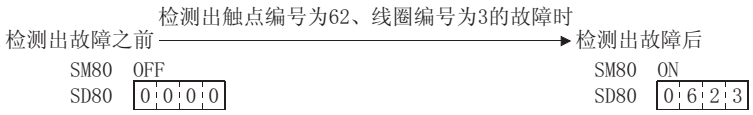
CHKST 指令为 ON 时，执行 CHK 指令。



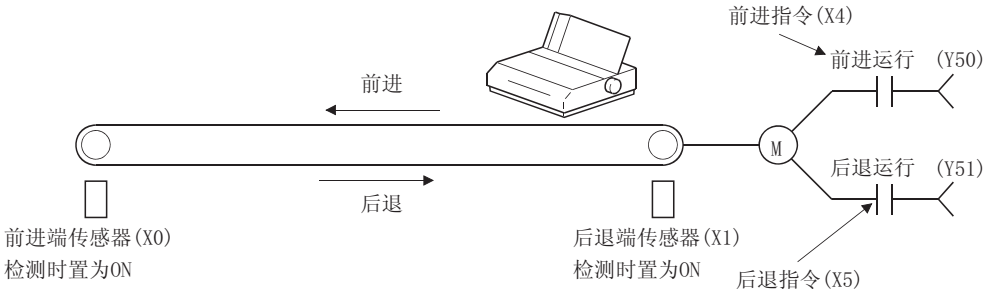
CHK

(1) CHK 指令是用于对如下所示的进行往复运动的系统的故障内容进行确认的指令。

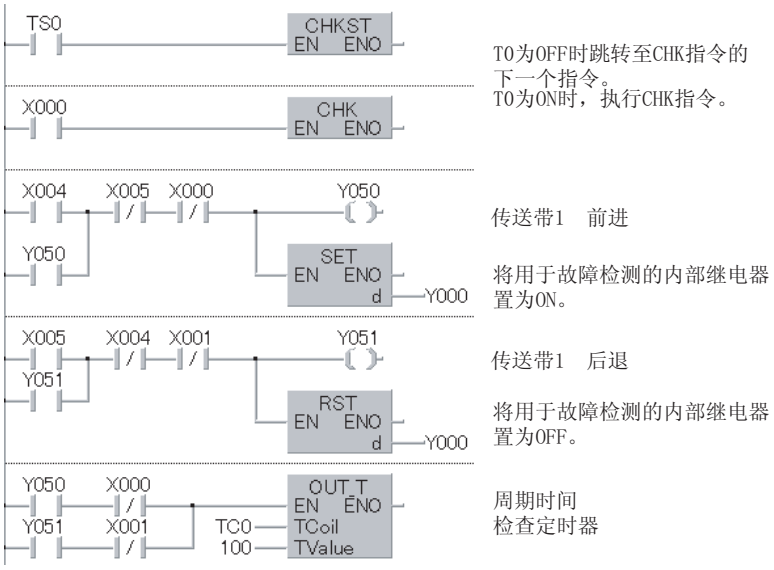
(a) 如果执行 CHK 指令，将进行指定检查条件的故障诊断检查，检测出故障时将 SM80 置为 ON，将故障编号以 BCD 值存储到 SD80 中。
此外，检测出故障时将发生出错代码为 9010 的出错。
SD80 的高 3 位中存储检测出故障的触点编号 [参阅 (3)]，低 1 位中将存储检测出故障的线圈编号 [参阅 (2)]。



(b) 对于位于 CHK 指令前面的触点指令，不是用于对 CHK 指令的执行进行控制，而是用于设置检查条件。



(c) 对上图的系统周期时间进行溢出检查时，创建如下所示的梯形图。



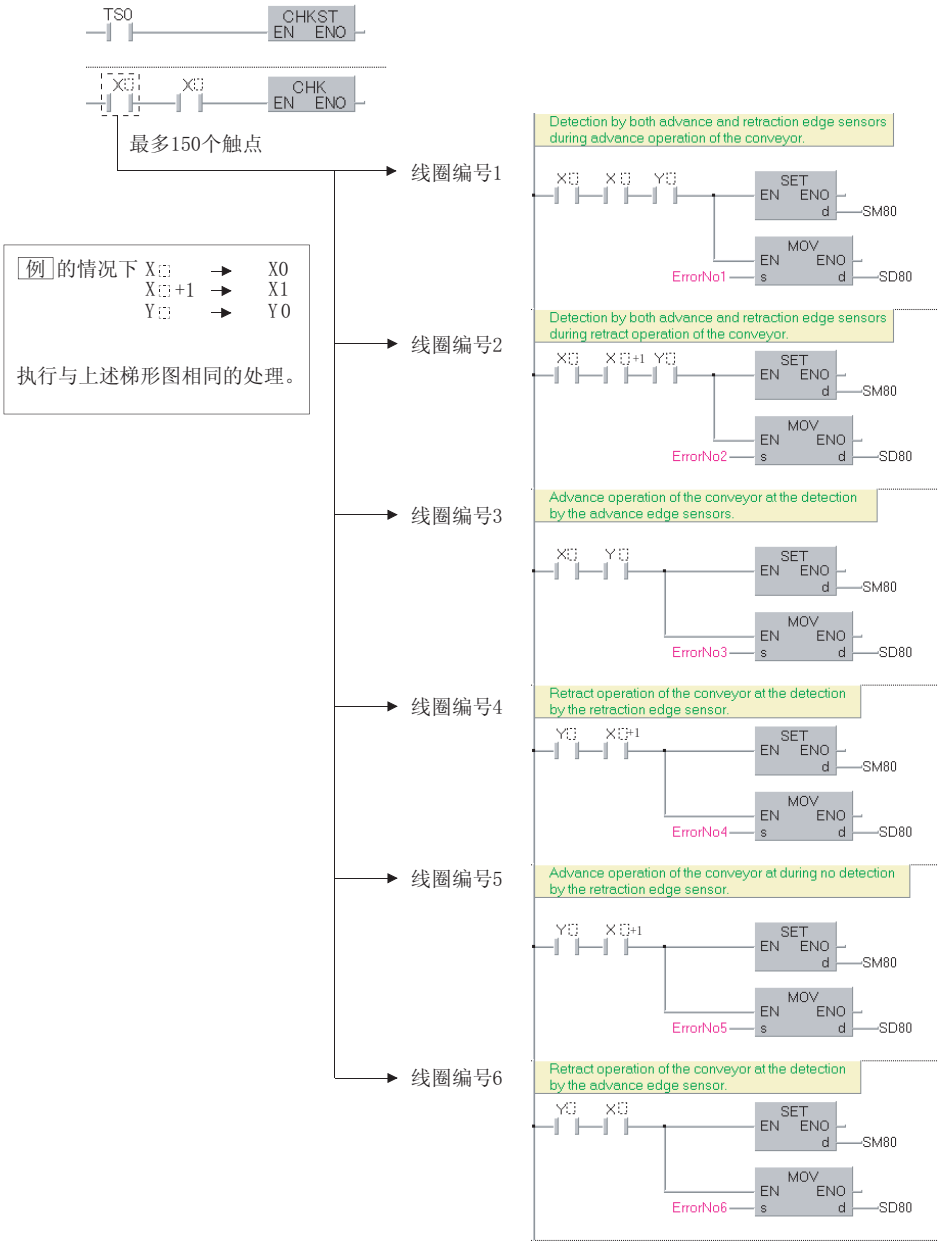
(d) 创建使用了 CHK 指令的梯形图时的注意事项如下所示。

1) 前进端传感器的触点与后退端传感器的触点编号 (X□) 必须连续。此外，应将前进端传感器的触点编号 (X□) 设置为小编号。

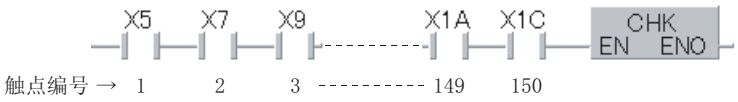
2) 对于与前进端传感器的触点编号 (X□) 为同一编号的输出 (Y□)*1，应按如下方式进行控制。
前进运行时 ..... 置为 ON
后退运行时 ..... 置为 OFF

*1 : 输出 (Y□) 被作为内部继电器使用，不能对外部进行输出。

(2) CHK 指令根据指定的触点，执行与下述梯形图相同的处理。



(3) 检测出故障时的触点编号从左侧的纵母线开始依次分配为 1 ～ 150。



(4) 执行 CHK 指令的情况下，应预先对 SM80、SD80 进行复位。
此外，执行 CHK 指令后，未对 SM80、SD80 进行复位的情况下，将无法再次执行 CHK 指令。
(SM80、SD80 的内容在用户复位之前将被保持。)

(5) CHK 指令的前面需要有 CHKST 指令。
CHK 指令与 CHKST 指令之间存在有除 LD、LDI、AND、ANI 指令以外的指令时，将变为出错状态。 (出错代码： 4235)

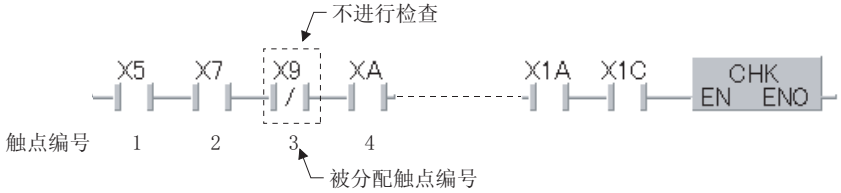
(6) CHK 指令可被写入到程序中的任意步中。
但是，CHK 指令的使用数有下述限制。

• 在执行的所有的程序文件中最多为 2 处。

• 1 个程序文件中仅为 1 处。

CHK 指令的使用超出了上述限制的情况下将变为出错状态。 (出错代码： 4235)

(7) 在 CHK 指令的前面应通过 LD、AND 指令设置检查条件。
不能通过其它触点指令设置检查条件。
通过 LDI、ANI 指令设置了检查条件的情况下，将不执行该检查条件的相关处理。
但 LDI、ANI 指令也将被分配故障检测时的触点编号。



(8) 根据 SM710 的 ON/OFF 状态故障检测方法有所不同。

(a) SM710 为 OFF 的情况下，按触点顺序对线圈编号 1 ～ 6 进行检查。
执行 CHK 指令时，对触点编号 1 的线圈编号 1 ～ 6 按顺序进行检查。在检查到触点编号 1 的线圈编号 6 时，对触点编号 2 的线圈编号 1 ～ 6 按顺序进行检查。
检查至触点编号 n 的线圈编号 6 时，结束 CHK 指令。

(b) SM710 为 ON 的情况下，按线圈顺序对触点编号 1 ～ n 进行检查。
如果执行 CHK 指令，将线圈编号 1 的梯形图按触点编号 1 ～ n 的顺序进行检查。检查至线圈编号 1 的梯形图的触点编号 n 时，对线圈编号 2 的梯形图按触点编号 1 ～ n 的顺序进行检查。
检查至线圈编号 6 的触点编号 n 时，结束 CHK 指令。

(9) 检测出多个故障的情况下，最先检测到的故障编号将被存储。
后检测到的故障编号将被忽略。

(10) 在低速执行类型程序中不能使用 CHK 指令。
将记述了 CHK 指令的程序文件设置到低速执行类型程序中的情况下，将变为运算出错状态，CPU 模块将停止运算。

出 错

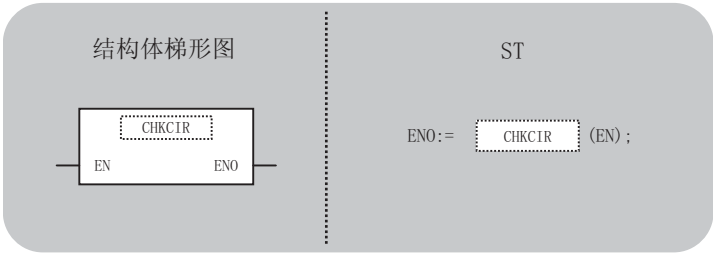
在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 存在有并联梯形图的情况下。 (出错代码： 4235)
- 触点指令超过了 150 个的情况下。 (出错代码： 4235)
- 执行了 CHKST 指令之后，未执行 CHK 指令时。 (出错代码： 4235)
- 未执行 CHKST 指令的状况下执行了 CHK 指令时。 (出错代码： 4235)
- 在低速执行类型程序中使用了 CHKST、CHK 指令时。 (出错代码： 4235)
- CHK 指令与 CHKST 指令之间存在有除 LD、LDI、AND、ANI 指令以外的指令时。 (出错代码： 4235)
- 在执行的所有程序文件中在 3 处以上位置使用了 CHK 指令时。 (出错代码： 4235)
- 在 1 个程序文件中在 2 处以上位置使用了 CHK 指令时。 (出错代码： 4235)

7.10.2 检查指令的检查格式变更

CHKCIR, CHKEND

CHKCIR
CHKEND



□ 中放入下述指令。

CHKCIR
CHKEND

输入自变量， EN: 执行条件 ：位
输出自变量， ENO: 执行结果 ：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
–	–								

功 能

CHKCIR,CHKEND

(1) 将 CHK 指令中使用的检查梯形图模式变更为任意的格式。
实际的故障检查是通过 CHKST、CHK 指令进行。

(2) 进行故障检查时，通过 CHK 指令中指定的检查条件及 CHKCIR ～ CHKEND 指令之间记述的梯形图模式进行。

备 注

关于 CHKST、CHK 指令，请参阅 7.10.1 项。

要 点

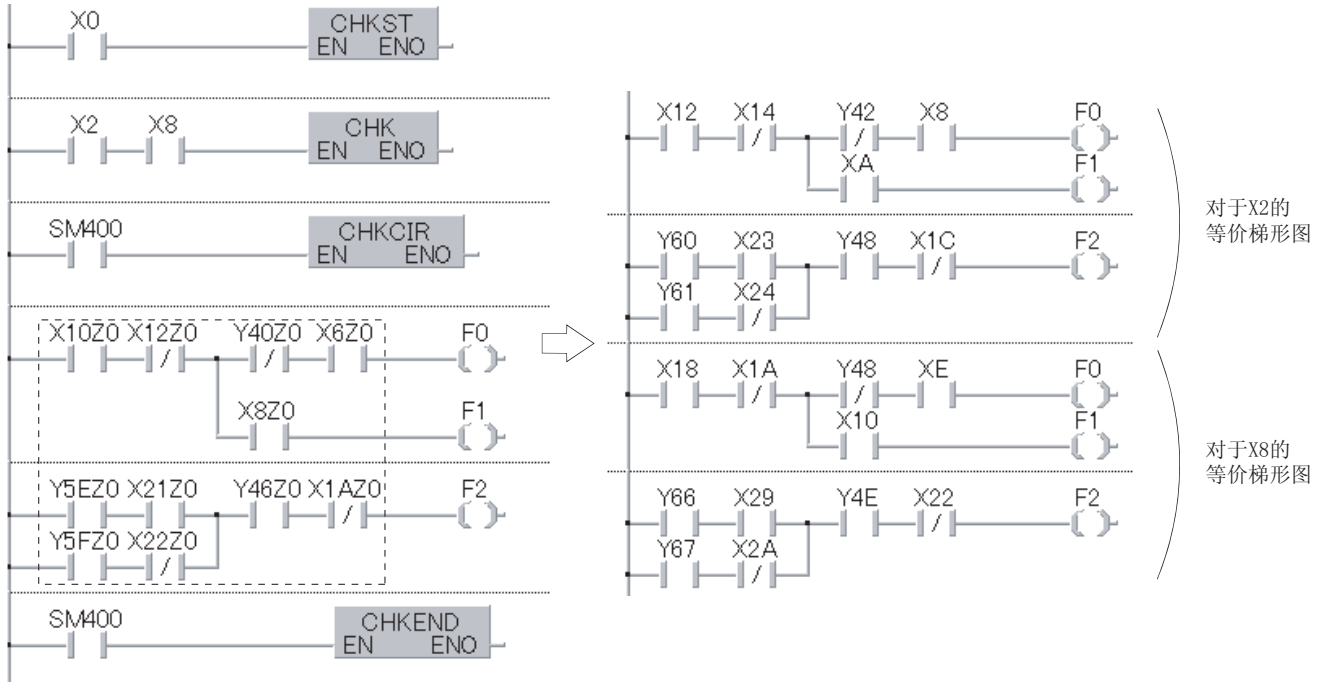
使用 CHKCIR ～ CHKEND 对 CHK 指令的检查格式进行变更时，应由用户创建附加了变址修饰 (Z0) 的梯形图。

(a) 检查条件（下图的 X2、X8）中所示的软元件编号，将成为梯形图模式中记述的各软元件编号（报警器 (F) 除外）的变址修饰值。
例）下图⬚中的 X10 的情况如下所示。

对应于检查条件 X2 时 ..... 以 X12
对应于检查条件 X8 时 ..... 以 X18
} 进行处理。

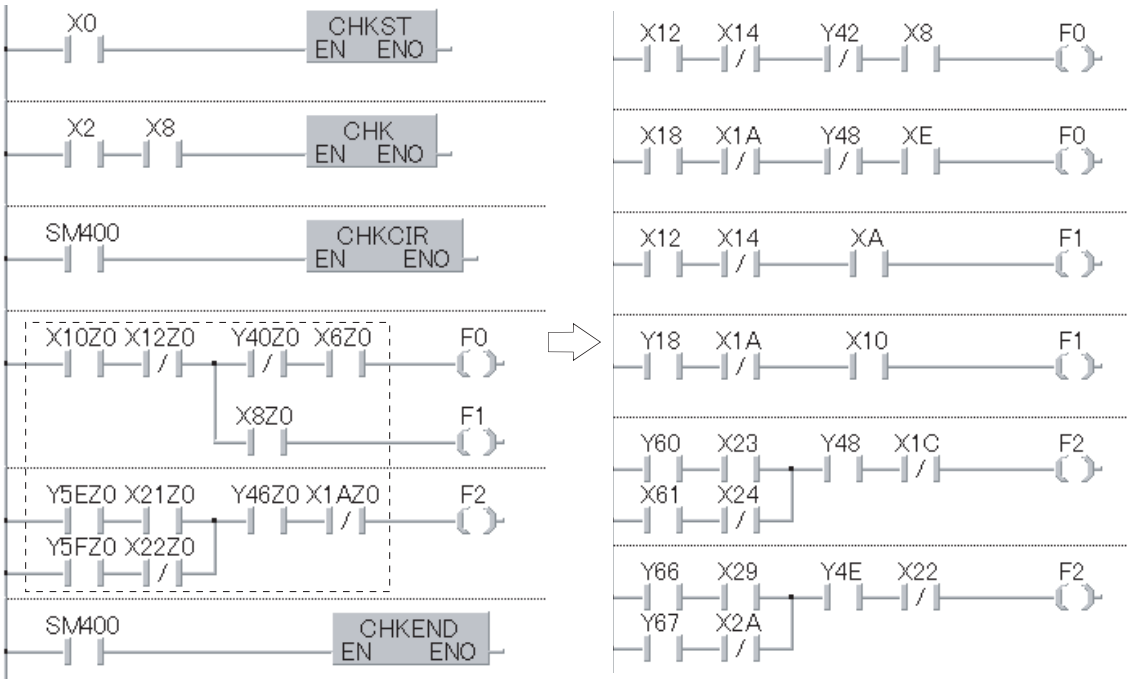
但是，根据 SM710 的 ON/OFF 状态故障检测顺序有所不同。
1) SM710 为 OFF 时，按触点顺序从线圈编号 1 开始依次进行检查。

[CHKCIR ～ CHKEND 中的指定梯形图]　　[CPU 模块中的检查顺序]



2) SM710 为 ON 时，按线圈顺序从触点编号 1 开始依次进行检查。

[CHKCIR ～ CHKEND 中的指定梯形图]　　[CPU 模块中的检查顺序]



(b) 进行故障检查时，根据各检查条件各自的梯形图模式，对 OUT F□的 ON/OFF 状态进行检查。
在全部检查条件中，如果梯形图模式中的 OUT F□有 1 个为 ON，则将 SM80 置为 ON。
此外，将变为 ON 的 OUT F□所对应的出错编号（触点编号及线圈编号）以 BCD 顺序存储到 SD80 中。

(c) 在梯形图模式中可使用的指令如下所示。
触点 ..... LD,LDI,AND,ANI,OR,ORI,ANB,ORB,MPS,MPP,MRD, 比较运算指令
线圈 ..... OUT F□

(d) 在梯形图模式的触点中可使用的软元件如下所示。
输入 (X)、输出 (Y)

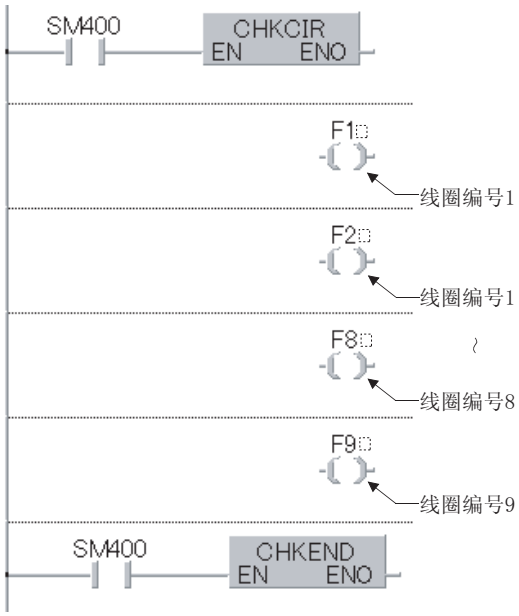
(e) 在梯形图模式的线圈中可使用的软元件仅为报警器 (F)。
但是，由于使用虚拟报警器 (F)，因此可以设置任意的值。
此外，即使重复也没有关系。

(f) 即使将与 CHK 指令中中使用的报警器 (F) 相同编号的报警器 (F) 用于 CHK 指令以外时也可正常进行 ON/OFF 控制。
将 CHK 指令中与 CHK 指令以外分别进行处理。

(g) 对于 CHK 指令中使用的报警器 (F)，实际上不进行 ON/OFF，因此即使通过外围设备进行监视也不进行 ON/OFF。

(h) 梯形图模式最多可创建 256 步。
此外，OUT F□最多可使用 9 个线圈。

(3) CHKCIR ～ CHKEND 中指定的梯形图的线圈编号按从上至下分配为线圈编号 1 ～ 9。



(4) CHKCIR、CHKEND 指令可被写入到程序中的任意步中。
在执行的所有的程序文件中最多可以在 2 处使用。
但是，1 个程序文件中只能在 1 处使用 CHK 指令。

(5) CHKCIR、CHKEND 指令不能用于低速执行类型程序。
将记述了 CHKCIR、CHKEND 指令的程序文件设置到低速执行类型程序的情况下，将变为运算出错状态。此外，高性能型 QCPU 的情况下将停止运算。

出 错

在以下情况下将变为运算出错状态，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- CHKCIR ～ CHKEND 指令在全部程序文件中存在有 3 处以上时。 (出错代码： 4235)
- CHKCIR ～ CHKEND 指令在 1 个程序文件中存在有 2 处以上时。 (出错代码： 4235)
- 执行 CHKCIR 指令后，未执行 CHKEND 指令时。 (出错代码： 4230)
- 在未执行 CHKCIR 指令的状况下执行了 CHKEND 指令时。 (出错代码： 4230)
- CHKST、CHK 指令被用于低速执行类型程序中时。 (出错代码： 4235)
- 梯形图模式中有 10 个以上的 F 时。 (出错代码： 4235)
- 梯形图模式超过了 257 步时。 (出错代码： 4235)
- 梯形图模式中存在有不能使用的软元件时。 (出错代码： 4235)
- 对梯形图模式的软元件进行了变址修饰时。 (出错代码： 4235)

7.11 字符串处理指令

7.11.1 BIN16 位 /32 位→ 10 进制 ASCII 转换

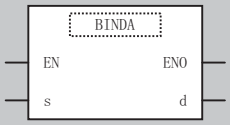
BINDA, DBINDA

Basic | High performance | Universal | L CPU

BINDA(P)
DBINDA(P)

P: 执行条件 : ↰

结构体梯形图



ST

```
ENO:= BINDA (EN, s, d);
```

中放入下述指令。

BINDA	BINDAP
DBINDA	DBINDAP

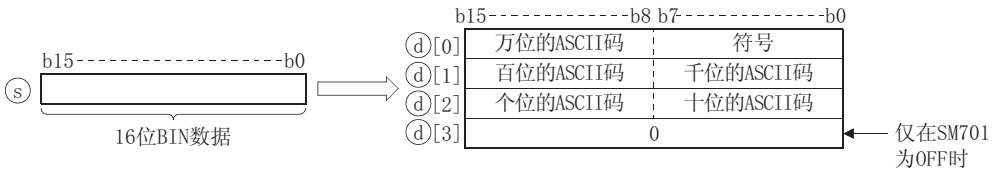
输入自变量,	EN:	执行条件	: 位
	s:	进行 ASCII 转换的 BIN 数据	: ANY16/32
输出自变量,	ENO:	执行结果	: 位
	d:	转换结果	: ANY16 的数组 (0..3)/(0..5) : 字符串 (8)/(12)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
ⓓ	-	○		-					-

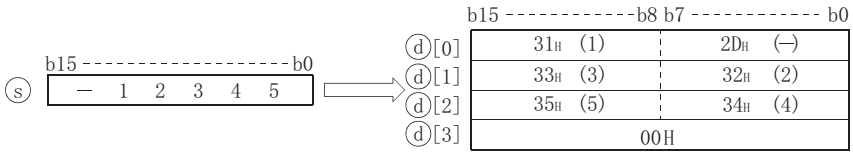
功 能

BINDA(P)

(1) 将ⓢ中指定的 BIN16 位数据以 10 进制数表示时的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定了 -12345 的情况下，在ⓓ以后按如下方式存储。

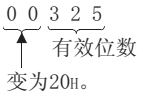


(2) ⓢ中可指定的 BIN 数据的范围为 -32768 ～ 32767。

(3) ⓓ中存储的运算结果情况如下所示。

(a) 在符号中，BIN 数据为正时存储 20H，为负时存储 2DH。

(b) 在有效位数左侧的 0 中存储 20H。（零限点。）



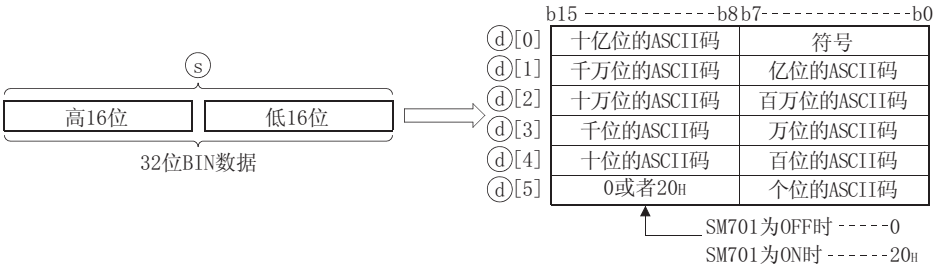
(c) 对于至ⓓ[3]中指定的软元件的数据存储，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。

SM701 为 OFF 时... 存储 0。

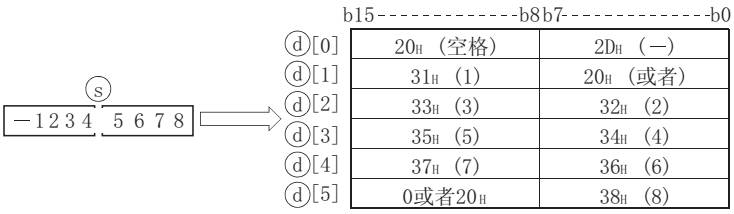
SM701 为 ON 时.... 不变化。

DBINDA(P)

(1) 将ⓢ中指定的 BIN32 位数据以 10 进制数表示时的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定了 -12345678 的情况下，在ⓓ以后按如下方式存储。

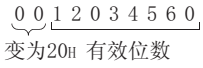


(2) 将ⓢ中可指定的 BIN 数据的范围为 -2147483648 ～ 2147483647。

(3) ⓓ中存储的运算结果情况如下所示。

(a) 在符号中，BIN 数据为正时存储 20H，为负时存储 2DH。

(b) 在有效位数左侧的 0 中存储 20H。（零限点。）



(c) 对于ⓓ[5]中指定的软元件的高 8 位中存储的数据，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。

SM701 为 OFF 时.... 存储 0。

SM701 为 ON 时..... 存储 20H。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

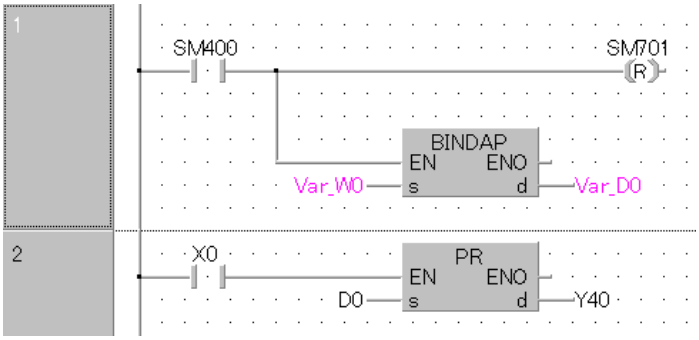
- ⓓ中指定的软元件超出了相应软元件的范围时。（通用型 QCPU、LCPU 时）

（出错代码： 4101）

程序示例

(1) 以下为将 16 位 BIN 数据的 Var_W0 的值通过 PR 指令以 10 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配到 D0 软元件中。）

[结构体梯形图]

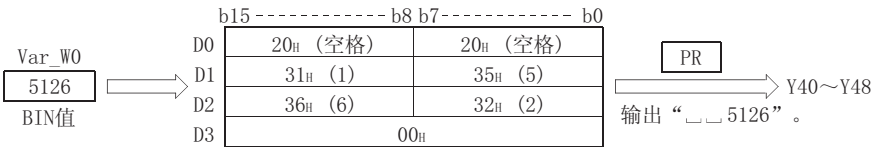



```
[ST]
IF SM400 THEN
     RST(TRUE,SM701);
     BINDAP(TRUE,Var_W0,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

[动作]

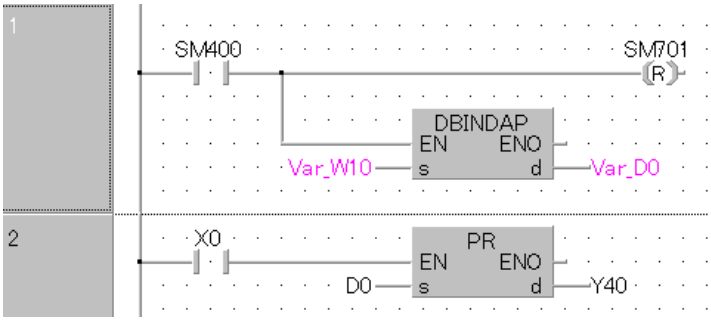
如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。

由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



(2) 以下为将 32 位 BIN 数据 Var_W10 的值通过 PR 指令以 10 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配至 D0 软元件中。）

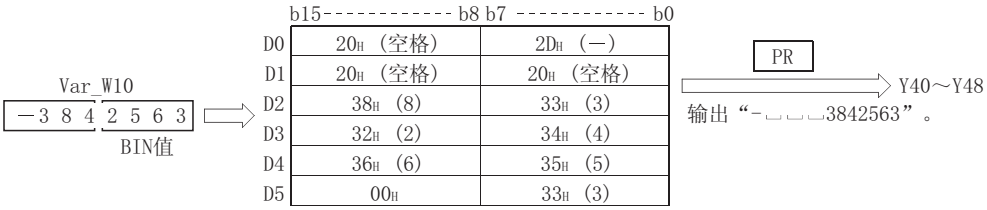
[结构体梯形图]




```
[ST]
IF SM400 THEN
     RST(TRUE,SM701);
     DBINDAP(TRUE,Var_W10,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

[动作]

如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。
由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



7.11.2 BIN16 位 /32 位→ 16 进制 ASCII 转换

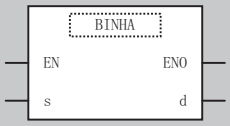
BINHA, DBINHA

Basic | High performance | Universal | L CPU

BINHA(P)
DBINHA(P)

P: 执行条件 : ⏎

结构体梯形图



ST

ENO:= BINHA (EN, s, d);

[ ] 中放入下述指令。

BINHA	BINHAP
DBINHA	DBINHAP

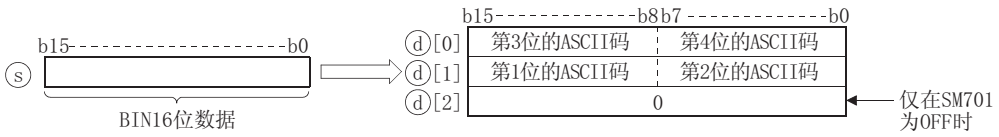
输入自变量， EN: 执行条件 : 位
s: 进行 ASCII 转换的 BIN 数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 转换结果 : ANY16 的数组 (0..2)/(0..4)
: 字符串 (6)/(10)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
ⓓ	-	○		-					-

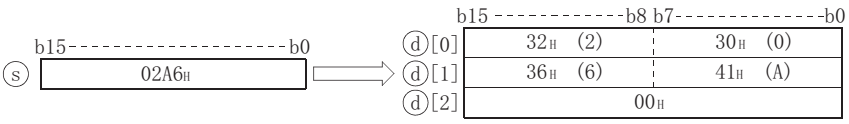
功 能

BINHA(P)

(1) 将ⓢ中指定的 BIN16 位数据以 16 进制数表示时的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定了 02A6H 的情况下，按以下方式存储到ⓓ以后。



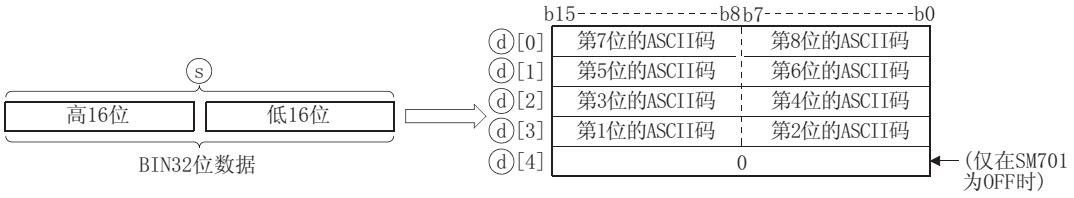
(2) ⓢ中可指定的 BIN 数据的范围为 0H ～ FFFFH。

(3) ⓓ中存储的运算结果被作为 4 位的 16 进制数处理。
因此，有效位数左侧的 0 被作为 0 处理。（不执行零限点。）

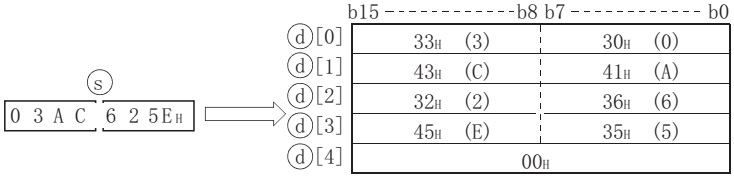
(4) 对于至ⓓ[2]中指定的软元件的数据存储，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。
SM701 为 OFF 时 ... 存储 0。
SM701 为 ON 时 .... 不变化。

DBINHA(P)

(1) 将ⓢ中指定的 BIN32 位数据以 16 进制数表示时的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定了 03AC625EH 的情况下，按以下方式存储到ⓓ以后。



(2) ⓢ中可指定的 BIN 数据的范围为 0H ～ FFFFFFFFH。

(3) ⓓ中存储的运算结果被作为 8 位的 16 进制数处理。
因此，有效位数左侧的 0 被作为 0 处理。（不进行零限点。）

(4) 对于至ⓓ[4]中指定的软元件的数据存储，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。
SM701 为 OFF 时 ... 存储 0。
SM701 为 ON 时 .... 不变化。

出 错

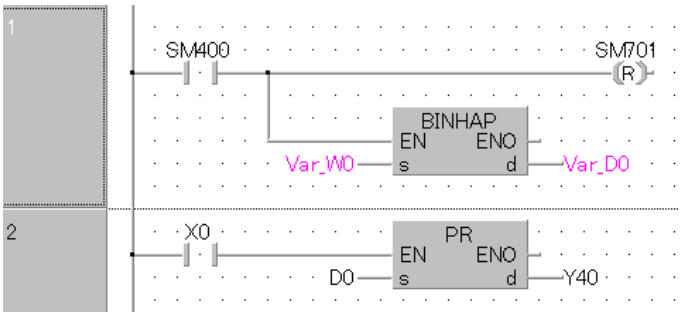
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ中指定的软元件超出了相应软元件的范围时。（仅通用型 QCPU）
（出错代码： 4101）

程序示例

(1) 以下为将 16 位 BIN 数据的 Var_W0 的值通过 PR 指令以 16 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配到 D0 软元件中。）

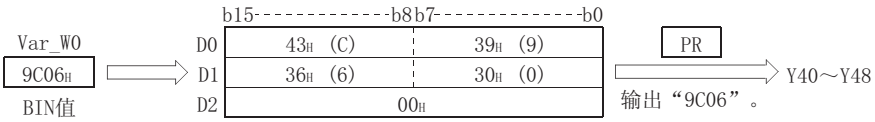
[结构体梯形图]




```
[ST]
IF SM400 THEN
     RST(TRUE,SM701);
     BINHAP(TRUE,Var_W0,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

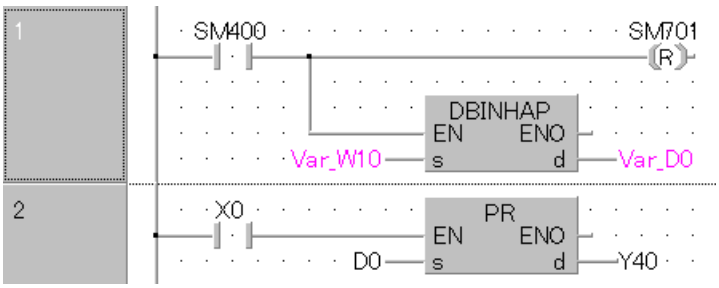
[动作]

如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。
由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



(2) 以下为将 32 位 BIN 数据的 Var_W10 的值通过 PR 指令以 16 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配至 D0 软元件中。）

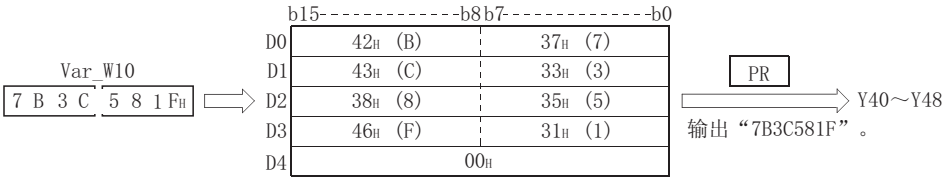
[结构体梯形图]




```
[ST]
IF SM400 THEN
    RST(TRUE,SM701);
    DBINHAP(TRUE,Var_W10,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

[动作]

如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。
由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



7.11.3 BCD4 位 /8 位→ 10 进制 ASCII 转换

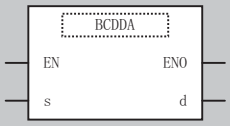
BCDDA, DBCDDA

Basic High performance Universal L CPU

BCDDA(P)
DBCDDA(P)

P: 执行条件 : ↰

结构体梯形图



ST

```
ENO:= BCDDA (EN, s, d);
```

[ ] 中放入下述指令。

BCDDA	BCDDAP
DBCDDA	DBCDDAP

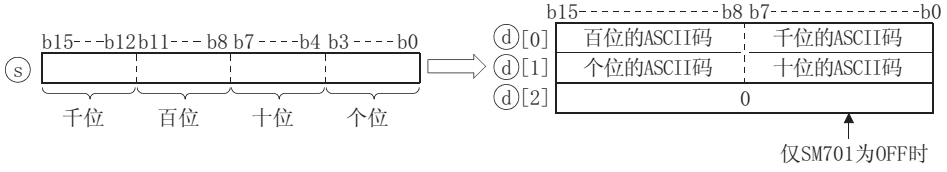
输入自变量， EN: 执行条件 : 位
s: 进行 ASCII 转换的 BCD 数据 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 转换结果 : ANY16 的数组 (0..2)/(0..4)
: 字符串 (6)/(10)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
ⓓ	-	○		-					-

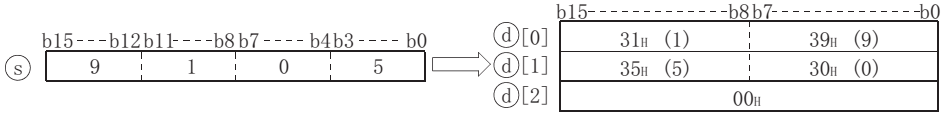
功 能

BCDDA(P)

(1) 将ⓢ中指定的 BCD4 位数据的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。

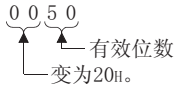


例如ⓢ中指定了 9105 的情况下，按以下方式存储到ⓓ以后。



(2) ⓢ中可指定的 BCD 数据的范围为 0 ～ 9999。

(3) 在ⓓ中存储的运算结果中，对有效位数左侧的 0 执行零限点。

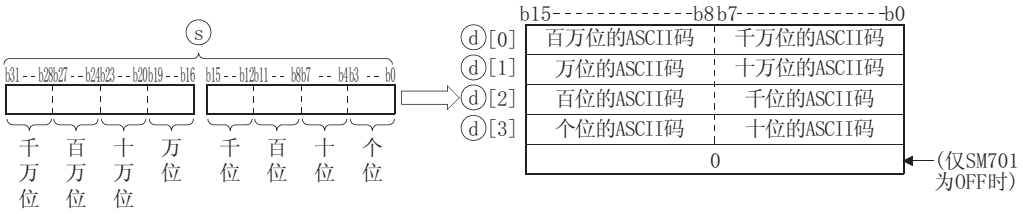


(4) 对于至ⓓ[2]中指定的软元件的数据的存储，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。

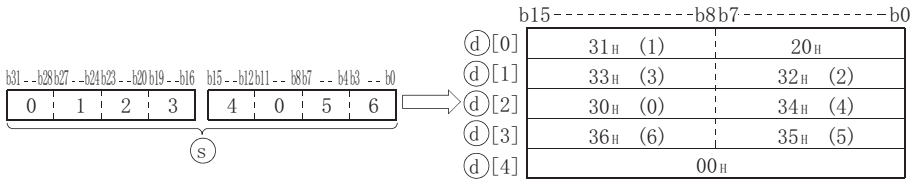
SM701 为 OFF 时 ... 存储 0。
SM701 为 ON 时 .... 不变化。

DBCDDA(P)

(1) 将ⓢ中指定的 BCD8 位数据的各个位的数值转换为 ASCII 码后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定了 01234056 的情况下，按以下方式存储到ⓓ以后。



(2) ⓢ中可指定的 BCD 数据的范围为 0 ～ 99999999。

(3) 在ⓓ中存储的运算结果中，对有效位数左侧的 0 执行零限点。

(4) 对于至ⓓ[4]中指定的软元件的数据的存储，根据 SM701（输出字符数切换信号）的 ON/OFF 状态而有所不同。

SM701 为 OFF 时 ... 存储 0。
SM701 为 ON 时 .... 不变化。

出 错

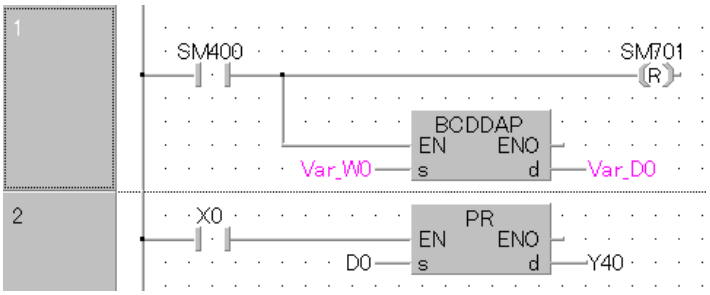
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- BCDDA(P) 指令时，ⓢ的数据超出了 0 ～ 9999 的范围时。 （出错代码： 4100）
- DBCDDA(P) 指令时，ⓢ的数据超出了 0 ～ 99999999 的范围时。 （出错代码： 4100）
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

(1) 以下位将 BCD4 位数据（Var_W0 的值）通过 PR 指令以 10 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配至 D0 软元件中。）

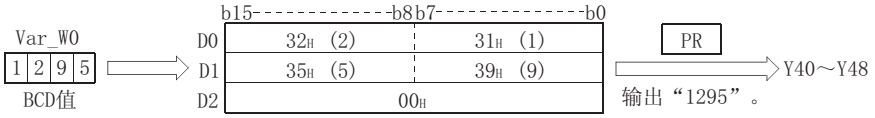
[结构体梯形图]




```
[ST]
IF SM400 THEN
     RST(TRUE,SM701);
     BCDDAP(TRUE,Var_W0,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

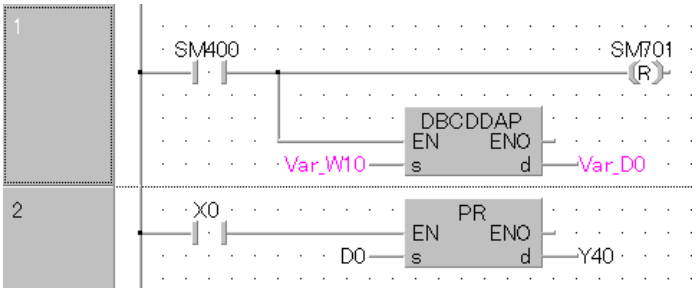
[动作]

如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。
由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



(2) 以下为将 BCD8 位数据（Var_W10 的值）通过 PR 指令以 10 进制数进行至 Y40 ～ Y48 的 ASCII 输出的程序。（将全局标签 Var_D0 分配至 D0 软元件中。）

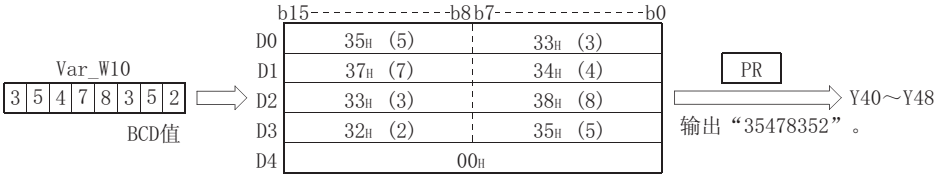
[结构体梯形图]




```
[ST]
IF SM400 THEN
    RST(TRUE,SM701);
    DBCDDAP(TRUE,Var_W10,Var_D0);
END_IF;
PR(X0,D0,Y40);
```

[动作]

如果将 X0 置为 ON，则通过 PR 指令进行至 Y40 ～ Y48 的 ASCII 输出。
由于 SM701 为 OFF，因此 PR 指令的输出至 ASCII 码 00H 为止。



7.11.4 10 进制 ASCII → BIN16 位 /32 位转换

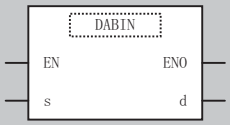
DABIN, DDABIN

Basic | High performance | Universal | L CPU

DABIN(P)
DDABIN(P)

P: 执行条件 : ⏋

结构体梯形图



ST

ENO:= DABIN (EN, s, d);

[ ] 中放入下述指令。

DABIN	DABINP
DDABIN	DDABINP

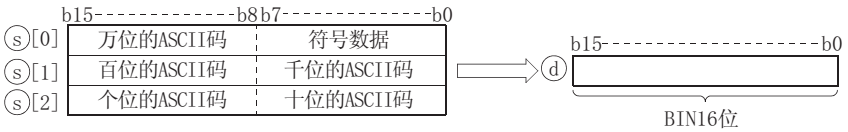
输入自变量， EN: 执行条件 : 位
s: 转换为 BIN 值的 ASCII 数据 : ANY16 的数组 (0..2)/(0..5)
: 字符串 (6)/(11)
输出自变量， ENO: 执行结果 : 位
d: 转换结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	-	○		-				○	-
ⓓ	○	○		○				-	-

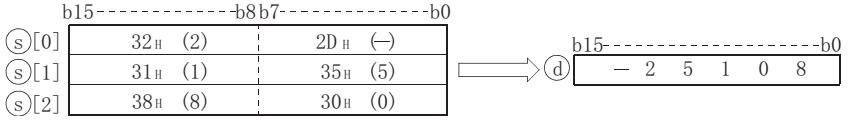
功 能

DABIN(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 10 进制 ASCII 数据转换为 BIN16 位数据后，存储到ⓓ中指定的软元件编号中。



例如ⓢ以后中指定了 -25108 的 ASCII 码的情况下，将按以下方式存储到ⓓ中。



(2) ⓢ[0] ～ ⓢ[2] 中可指定的 ASCII 数据的范围为 -32768 ～ 32767。

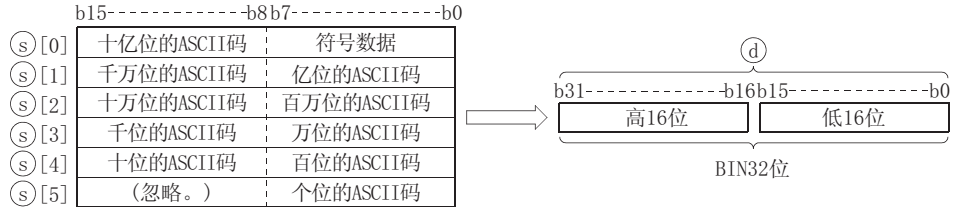
(3) 在符号数据中，转换数据为正时设置 20H，为负时设置 2DH。
（设置了除 20H,2DH 以外的情况下，将被作为正的数据处理。）

(4) 各个位中可设置的 ASCII 码的范围为 30H ～ 39H。

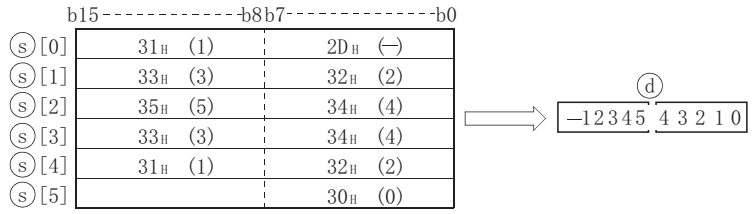
(5) 各个位中设置的 ASCII 码为 20H, 00H 时，将被作为 30H 处理。

DDABIN(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 10 进制 ASCII 数据转换为 BIN32 位数据后，存储到ⓓ中指定的软元件编号中。



例如ⓢ以后指定了 -1234543210 的 ASCII 码的情况下，按以下方式存储到ⓓ中。



(2) ⓢ[0] ～ ⓢ[5] 中可指定的 ASCII 数据的范围为 -2147483648 ～ 2147483647。此外，ⓢ[5] 的高位字节中存储的数据将被忽略。

(3) 在符号数据中，转换数据为正时设置 20H，为负时设置 2DH。
（设置了除 20H、2DH 以外的情况下，将被作为正的数据处理。）

(4) 各个位中可设置的 ASCII 码的范围为 30H ～ 39H。

(5) 各个位中设置的 ASCII 码为 20H、00H 时，将被作为 30H 处理。

出 错

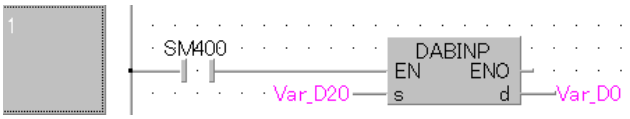
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ[0] ～ ⓢ[5] 中指定的各个位的 ASCII 码超出了 30H ～ 39H、20H、00H 以外时。
（出错代码： 4100）
- ⓢ[0] ～ ⓢ[5] 中指定的 ASCII 数据超出了下述范围时。（出错代码： 4100）
使用 DABIN(P) 指令时..... -32768 ～ 32767
使用 DDABIN(P) 指令时..... -2147483648 ～ 2147483647
- ⓢ中指定的软元件超出了相应软元件的范围时。
（通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

(1) 以下为将 Var_D20 中设置的符号及 10 进制 5 位数的 ASCII 数据转换为 BIN 值后，存储到 Var_D0 中的程序。

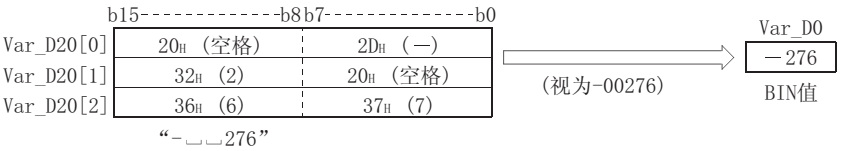
[结构体梯形图]



[ST]

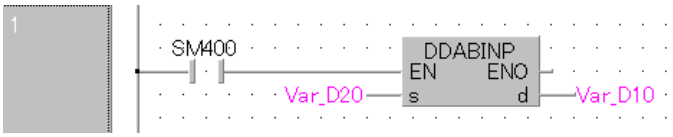
```
DABINP(SM400,Var_D20,Var_D0);
```

[动作]



(2) 以下为将 Var_D20 中设置的符号及 10 进制 10 位数的 ASCII 数据转换为 BIN 值后，存储到 Var_D10 中的程序。

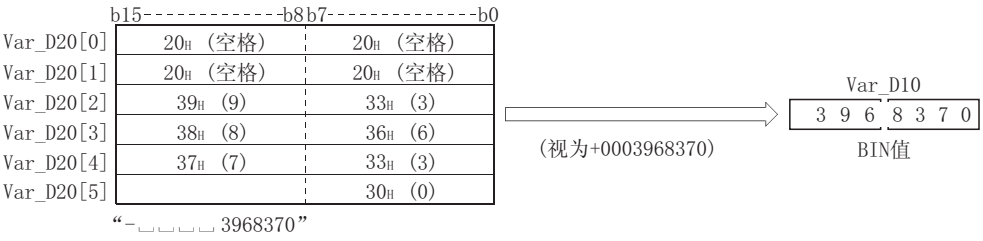
[结构体梯形图]



[ST]

```
DDABINP(SM400,Var_D20,Var_D10);
```

[动作]



7.11.5 16 进制 ASCII → BIN16 位 /32 位转换

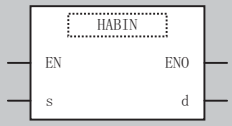
HABIN, DHABIN

Basic | High performance | Universal | L CPU

HABIN(P)
DHABIN(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= HABIN (EN, s, d);

中放入下述指令。

HABIN	HABINP
DHABIN	DHABINP

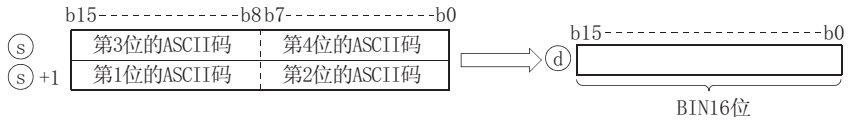
输入自变量， EN: 执行条件 : 位
s: 转换为 BIN 值的 ASCII 数据 : ANY16/32 的数组 (0..3)
: 字符串 (4)/(8)
输出自变量， ENO: 执行结果 : 位
d: 转换结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	–	○		–				○	–
ⓓ	○	○		○				–	–

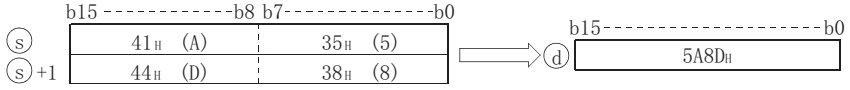
功 能

HABIN(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 16 进制 ASCII 数据转换为 BIN16 位数据后，存储到ⓓ中指定的软元件编号中。



例如ⓢ以后指定了 5A8DH 的 ASCII 码的情况下，将按以下方式存储到ⓓ中。

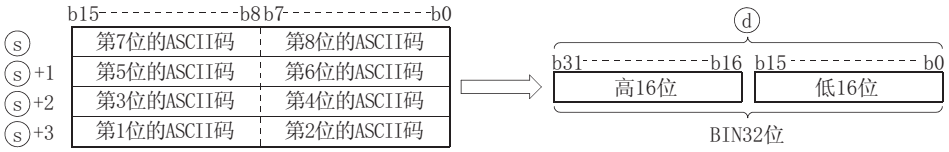


(2) ⓢ～ⓢ+1 中可指定的 ASCII 数据的范围为 0000H ～ FFFFH。

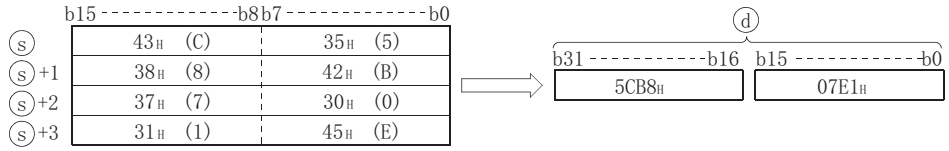
(3) 各位中可设置的 ASCII 码的范围为 30H ～ 39H 及 41H ～ 46H。

DHABIN(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 16 进制 ASCII 数据转换为 BIN32 位数据后，存储到ⓓ中指定的软元件编号中。



例如ⓢ以后中制定了 5CB807E1H 的 ASCII 码的情况下，将按以下方式存储到ⓓ+1, ⓓ中。



(2) ⓢ～ⓢ+3 中可指定的 ASCII 数据的范围为 00000000H ～ FFFFFFFFH。

(3) 各位中可设置的 ASCII 码的范围为 30H ～ 39H 及 41H ～ 46H。

出 错

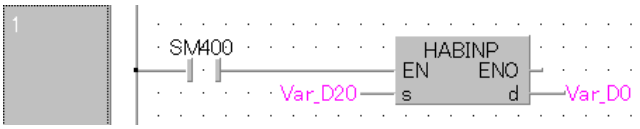
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ～ⓢ+3 中指定的各位的 ASCII 码超出了 30H ～ 39H, 41H ～ 46H 的范围时。
  (出错代码： 4100)
- ⓢ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

程序示例

(1) 以下为将 Var_D20 中设置的 16 进制 4 位的 ASCII 数据转换为 BIN 值后存储到 Var_D0 中的程序。

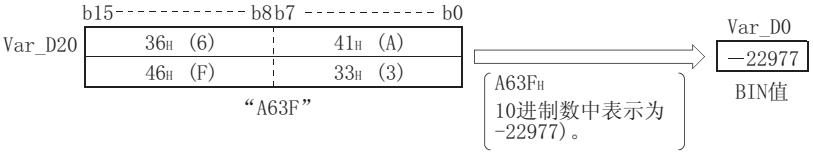
[结构体梯形图]



[ST]

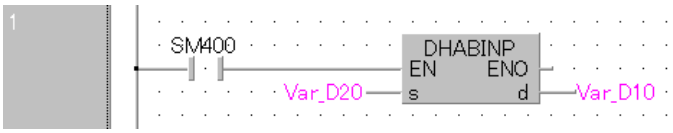
```
HABINP(SM400,Var_D20,Var_D0);
```

[动作]



(2) 以下为将 Var_D20 中设置的 16 进制 8 位的 ASCII 数据转换为 BIN 值后存储到 Var_D10 中的程序。

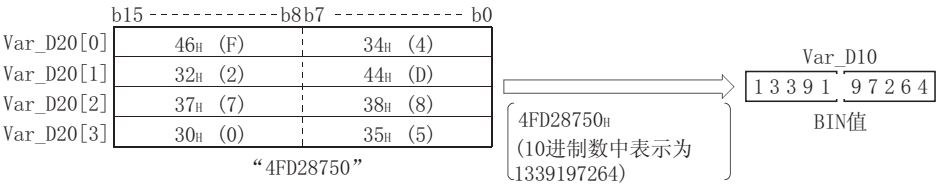
[结构体梯形图]



[ST]

```
DHABINP(SM400,Var_D20,Var_D10);
```

[动作]



7.11.6 10 进制 ASCII → BCD4 位 /8 位转换

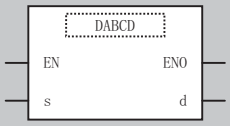
DABCD, DDABCD

Basic | High performance | Universal | L CPU

DABCD(P)
DDABCD(P)

P: 执行条件 :

结构体梯形图



ST

ENO:= DABCD (EN, s, d);

中放入下述指令。

DABCD	DABCDP
DDABCD	DDABCDP

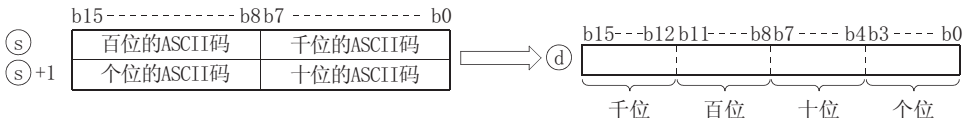
输入自变量, EN: 执行条件 : 位
s: 转换为 BCD 值的 ASCII 数据 : ANY16/32 的数组 (0..3)
: 字符串 (4)/(8)
输出自变量, ENO: 执行结果 : 位
d: 转换结果 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	-	○		-				○	-
ⓓ	○	○		○				-	-

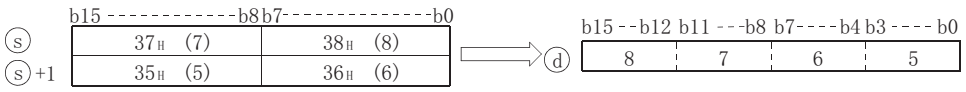
功 能

DABCD(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 10 进制 ASCII 数据转换为 BCD4 位数据后，存储到ⓓ中指定的软元件编号中。



例如ⓢ以后指定了 8765 的 ASCII 码的情况下，将按以下方式存储到ⓓ中。



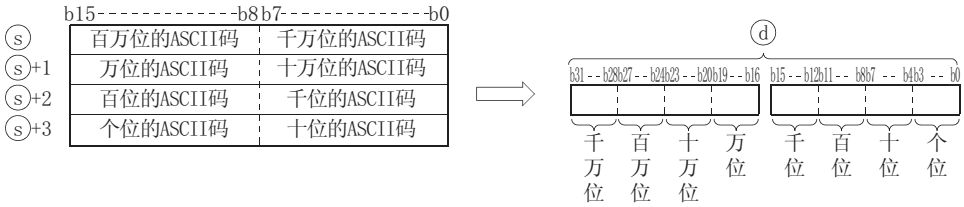
(2) ⓢ ～ ⓢ +1 中可指定的 ASCII 数据的范围为 0 ～ 9999。

(3) 各位中可设置的 ASCII 码的范围为 30H ～ 39H。

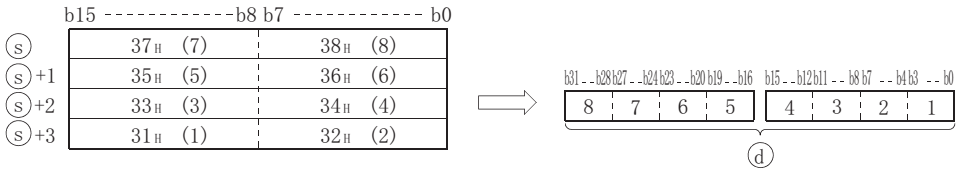
(4) 各位中设置的 ASCII 码为 20H，00H 时，将作为 30H 处理。

DDABCD(P)

(1) 将ⓢ中指定的软元件编号以后中存储的 10 进制 ASCII 数据转换为 BCD8 位数据后，存储到ⓓ中指定的软元件编号以后。



例如ⓢ以后中指定了 87654321 的 ASCII 码的情况下，将按以下方式存储到ⓓ中。



(2) ⓢ～ⓢ+3 中可指定的 ASCII 数据的范围为 0 ～ 99999999。

(3) 各位中可设置的 ASCII 码的范围为 30H ～ 39H。

(4) 各位中设置的 ASCII 码为 20H，00H 时，将作为 30H 处理。

出 错

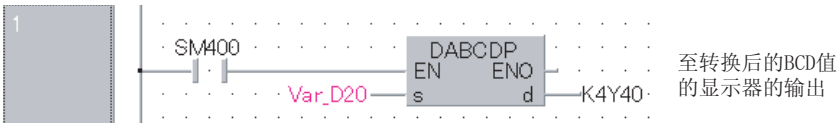
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ的数据中包含有除 0 ～ 9 以外的字符时。 (出错代码： 4100)
- ⓢ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

程序示例

(1) 以下为将 Var_D20 中设置的 10 进制数 ASCII 数据转换为 BCD4 位数据后输出到 Y40 ～ Y4F 中的程序。

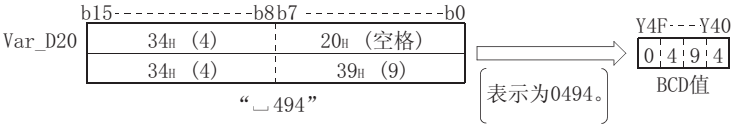
[结构体梯形图]



[ST]

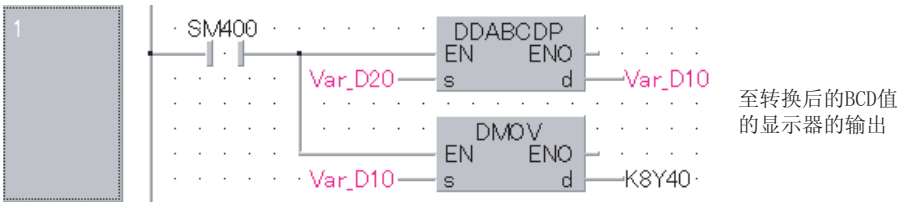
```
DABCDP(SM400,Var_D20,K4Y40);
```

[动作]



(2) 以下为将 Var_D20 中设置的 10 进制数 ASCII 数据转换为 BCD8 位数据后存储到 Var_D10 中的同时，输出到 Y40 ～ Y5F 中的程序。

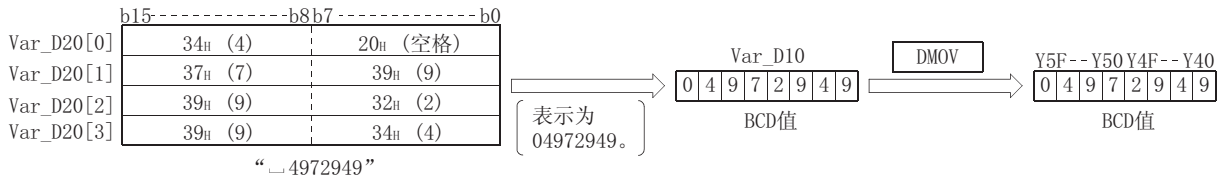
[结构体梯形图]



[ST]

```
IF SM400 THEN
    DDABCDP(TRUE,Var_D20,Var_D10);
    DMOV(TRUE,Var_D10,K8Y40);
END_IF;
```

[动作]



7.11.7 软元件的注释数据读取

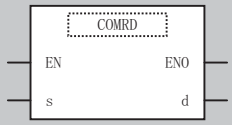
COMRD

Basic | High performance | Universal | L CPU

COMRD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= COMRD (EN, s, d);

中放入下述指令。

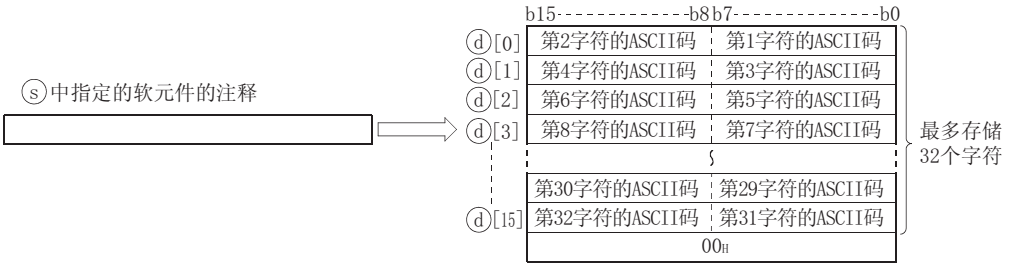
COMRD　　COMRDP

输入自变量，EN:　执行条件　：位
　　s:　读取的注释　：ANY_SIMPLE
输出自变量，ENO:　执行结果　：位
　　d:　读取的注释　：ANY16的数组(0..7)
　　：字符串(32)

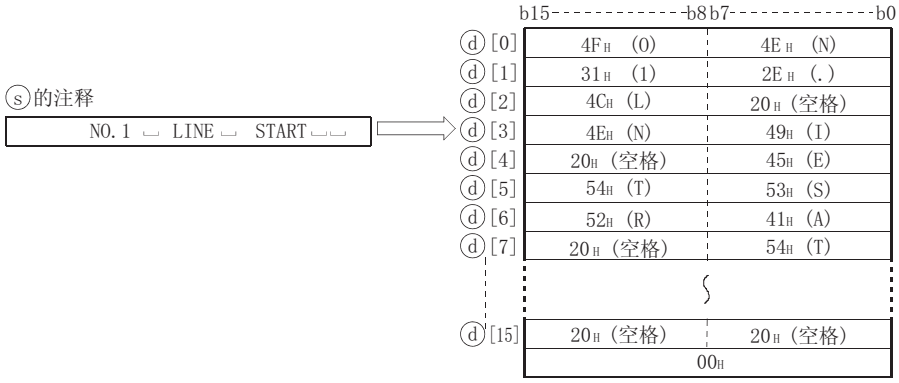
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	○	○		○			-		○
ⓓ	-	○		-			-		-

功 能

(1) 对ⓢ中指定的软元件编号的注释进行读取后，以ASCII码存储到ⓓ中指定的软元件编号以后。



例如ⓢ中指定的软元件的注释为 “No.1␣LINE␣START␣␣”的情况下，将按以下方式存储到ⓓ以后。



(2) ⓢ中指定的软元件编号被进行了注释范围设置，注释未被登录的情况下，将注释的字符均作为 20H(空格) 处理。

(3) 对于存储ⓓ的最终字符的软元件编号 +1，根据 SM701(输出字符数切换信号) 的 ON/OFF 状态而有所不同。

SM701 为 OFF 时....: 不变化。
SM701 为 ON 时.....: 存储 0。

(4) 读取注释时，指令结束后 SM720 将进行 1 个扫描 ON。
此外，指令执行过程中 SM721 将处于 ON 状态。
SM721 处于 ON 的过程中，不能执行 COMRD(P) 指令。如果执行将变为无处理。

要 点

1. COMRD(P) 指令中使用的软元件的注释使用存储卡中存储的注释文件。
   程序存储器中存储的注释文件不能使用。
2. COMRD(P) 指令中使用的注释文件是在参数模式的 “可编程控制器文件设置”中进行设置。
   未在可编程控制器文件设置中对使用的注释文件进行设置的情况下，将不能通过 COMRD(P) 指令输出软元件的注释。
3. 不要在中断程序中使用 COMRD(P) 指令。如果执行将导致误作动。

出 错

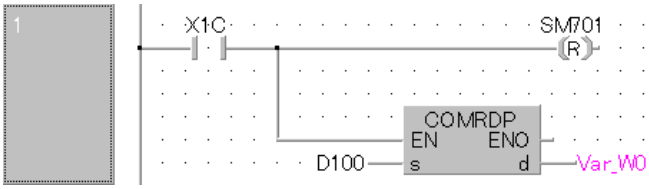
在以下情况下，将变为运算出错状态，出错标志将 ON。

- ⓢ中指定的软元件编号中，注释未被登录时。 (出错代码 : 4100)
- ⓓ中指定的软元件编号不是字软元件时。 (出错代码 : 4101)
- ⓓ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码 : 4101)

程序示例

以下为将 X1C 置为 ON 时，将 D100 中设置的注释以 ASCII 码存储到 Var_W0 以后的程序。

[结构体梯形图]




```
[ST]
IF X1C THEN
     RST(TRUE,SM701);
     COMRDP(TRUE,D100,Var_W0);
END_IF;
```

[动作]

D100的注释

ALINE ␣ TARGET

	b15----------b8	b7----------b0
Var_W0[0]	4CH (L)	41H (A)
Var_W0[1]	4EH (N)	49H (I)
Var_W0[2]	20H (空格)	45H (E)
Var_W0[3]	41H (A)	54H (T)
Var_W0[4]	57H (G)	52H (R)
Var_W0[5]	54H (T)	55H (E)
Var_W0[6]	20H (空格)	20H (空格)
Var_W0[7]	20H (空格)	20H (空格)
	⟆	
Var_W0[15]	20H (空格)	20H (空格)
Var_W0[16]	00H	

注意事项

(1) 数个扫描后处理结束。

(2) 指令结束前 (SM720 处于 ON 状态) 即使将 COMRD(P)/PRC 指令的启动信号 (执行指令) 置为 ON，也不能执行 COMRD(P)/PRC 指令。应将 COMRD(P)/PRC 指令设置为在 SM721 为 OFF 时执行。

(3) 不能同时访问 2 个以上的文件注释。

(4) 以下指令共享使用 SM721，因此不能同时执行。

指令名	执行中 ON	结束后 1 个扫描 ON	异常结束时 ON
SP_FREAD SP_FWRITE	SM721	通过指令指定	(指令中指定的软元件)+1
PRC COMRD(P)		SM720	无

7.11.8 字符串的长度检测

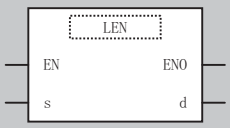
LEN

Basic　High performance　Universal　L CPU

LEN(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= LEN (EN, s, d);

中放入下述指令。

LEN　　LENP

输入自变量，EN：　执行条件　　：位

s：　字符串　　：ANY16

：字符串

输出自变量，ENO：　执行结果　　：位

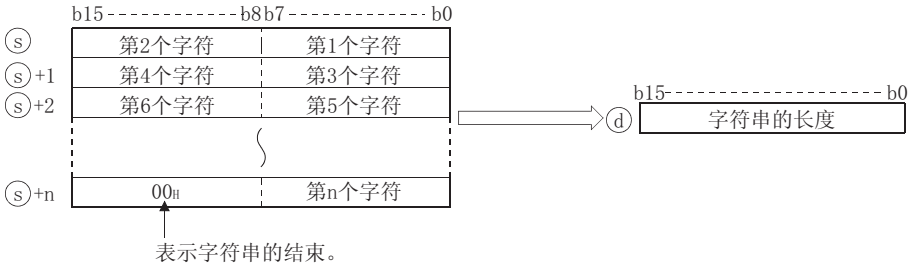
d：　存储检测的字符串长度的软元件编号　　：ANY16

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	－	○		－				○	－
ⓓ	○	○		○				－	－

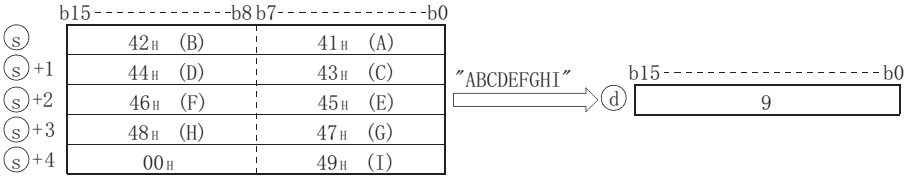
功 能

对ⓢ中指定的字符串的长度进行检测后，存储到ⓓ中指定的软元件编号以后。

将ⓢ中指定的软元件编号开始至存储了00H的软元件编号为止的数据作为字符串处理。



例如ⓢ以后存储了 “ABCDEFGHI” 的情况下，ⓓ中将存储 9。



出 错

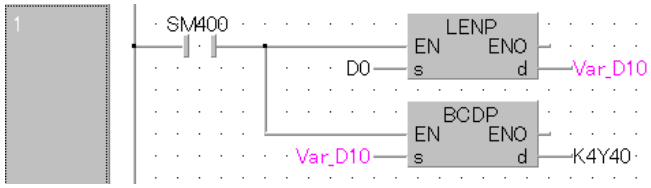
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的软元件编号以后，相应软元件范围内未设置 00H 时。 （出错代码： 4101）

程序示例

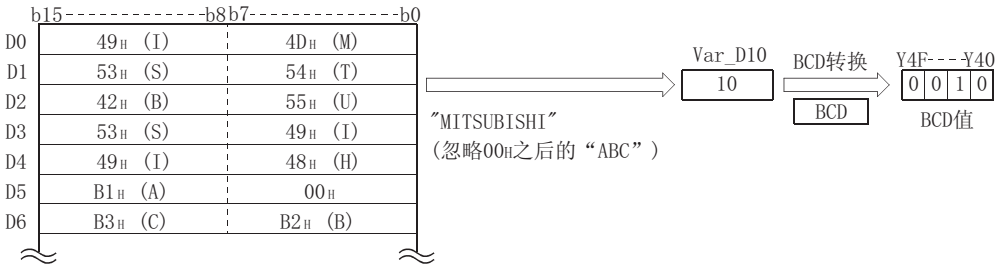
以下为将 D0 算起的字符串的长度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     LENP(TRUE,D0,Var_D10);
     BCDP(TRUE,Var_D10,K4Y40);
END_IF;
```

[动作]



7.11.9 BIN16 位 /32 位→字符串转换

STR, DSTR

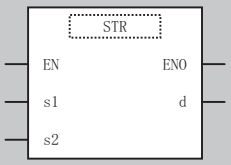
Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

STR(P)
DSTR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= STR (EN, s1, s2, d);

中放入下述指令。

STR	STRP
DSTR	DSTRP

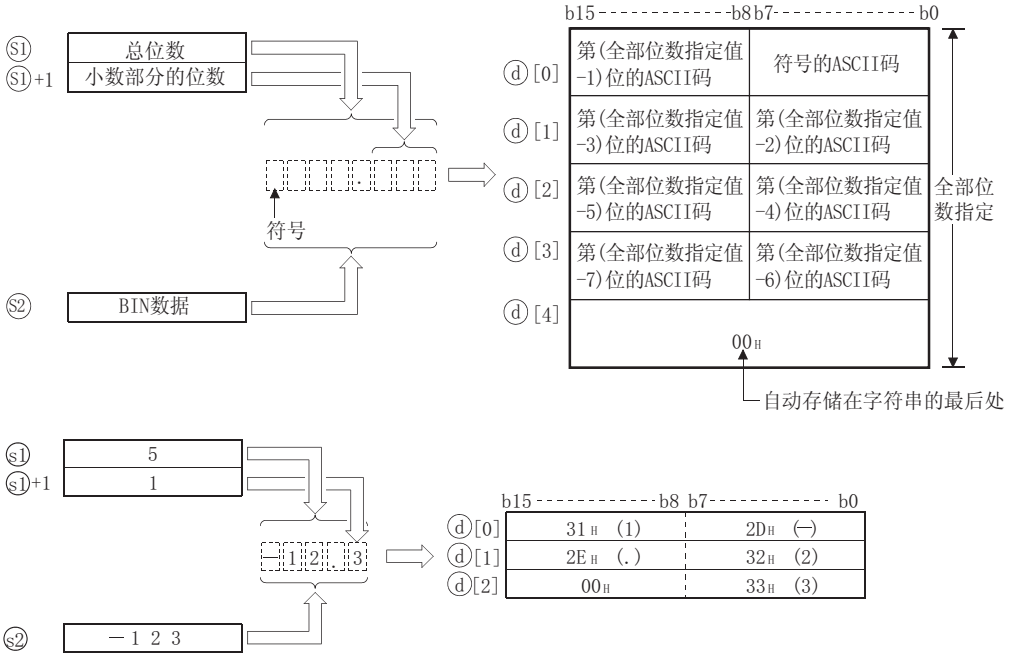
输入自变量，	EN:	执行条件	：位
	s1:	转换的数值的位数	：ANY32
	s2:	转换的 BIN 数据	：ANY16/32
输出自变量，	ENO:	执行结果	：位
	d:	转换后的字符串	：ANY16 的数组 (0..4)/(0..5) ：字符串 (9)/(14)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	○	○		○				-	-
ⓢ2	○	○		○				○	-
ⓓ	-	○		-				-	-

功 能

STR(P)

(1) 将ⓢ2 中指定的 BIN16 位数据在ⓢ1 中指定的位置处附加小数点后转换为字符串，存储到ⓓ 中指定的软元件编号以后。



(2) ⓢ1 中可指定的全部位数为 2 ～ 8 位。

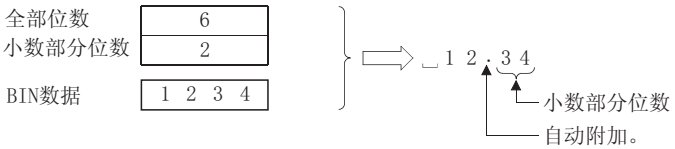
(3) ⓢ1 +1 中可指定的小数部分位数为 0 ～ 5 位。
但是，应设置为小数部分位数≤ (全部位数 -3)。

(4) ⓢ2 中可指定的 BIN 数据的范围为 -32768 ～ 32767。

(5) 对于转换后的字符串数据，按如下所示存储到ⓓ以后的软元件编号中。

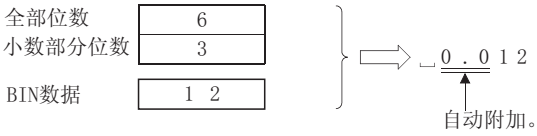
(a) 在符号中，BIN 数据为正时存储 20H(空格)，为负时存储 2DH(-)。

(b) 小数部分位数被设置为 0 以外的情况下，指定的位数 +1 位中将自动地存储 2EH(.)。

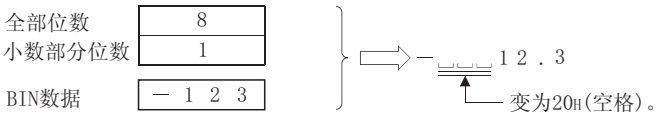


小数部分位数为 0 时，不存储 2EH(.)。

(c) 小数部分位数的值大于 BIN 数据的位数的情况下，将自动地附加 0 后进行转换，变为 0.□□□□□。



(d) 从全部位数的值中减去符号、小数点的位数大于 BIN 数据的位数的情况下，在符号与数值之间存储 20H(空格)。

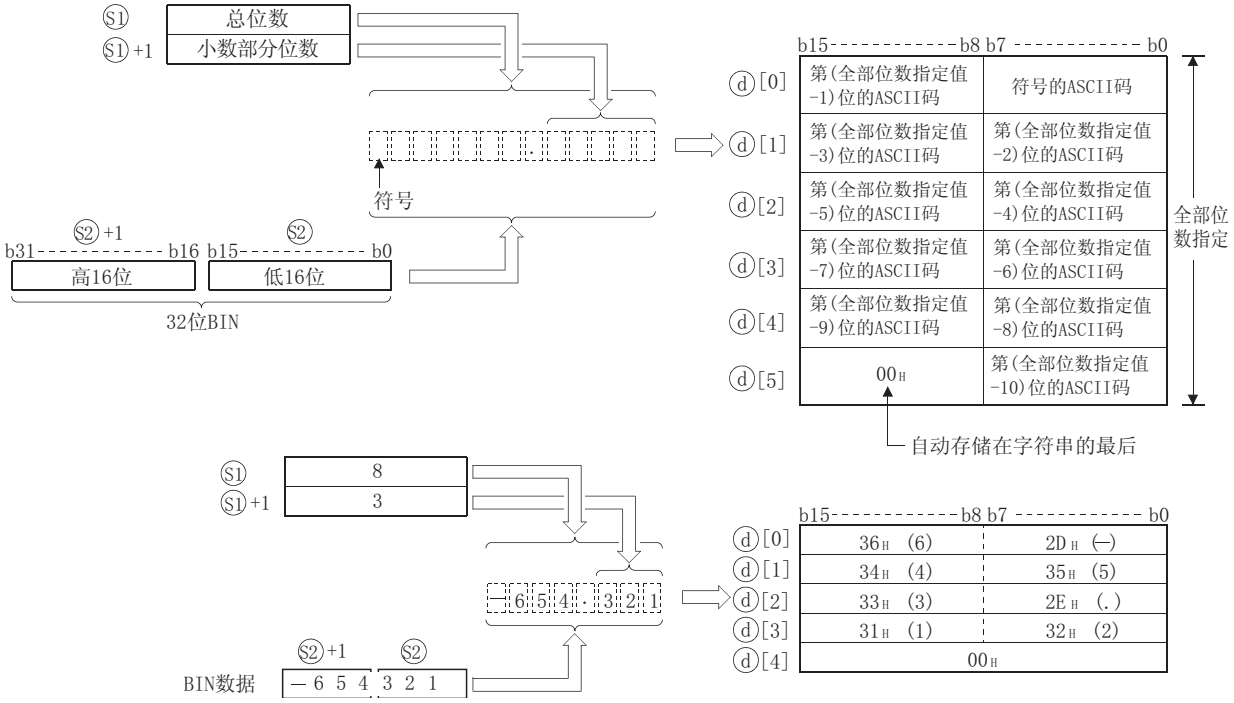


BIN 数据的位数大的情况下，将变为出错状态。

(e) 在转换后的字符串的最后处将自动地存储 00H。

DSTR(P)

(1) 将㉒中指定的 BIN32 位数据，在㉑中指定的位置处附加小数点并转换为字符串后，存储到ⓓ中指定的软元件编号以后。



(2) ㉑中可指定的全部位数为 2 ～ 13 位。

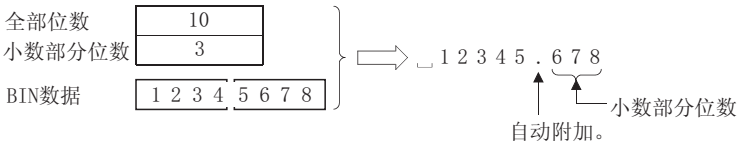
(3) ㉑+1 中可指定的小数部分位数为 0 ～ 10 位。
但是，应设置为小数部分位数≤（全部位数 -3)。

(4) ㉒中可指定的 BIN 数据的范围为 -2147483648 ～ 2147483647。

(5) 转换后的字符串数据按如下所示存储到ⓓ以后的软元件编号中。

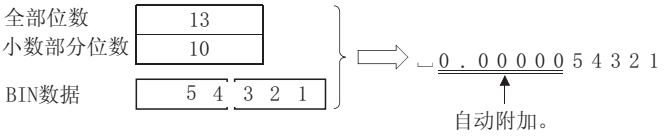
(a) 在符号中，BIN 数据为正时存储 20H(空格)，为负时存储 2DH(-)。

(b) 小数部分位数设置为 0 以外的情况下，在指定的位数 +1 位中将自动存储 2EH(.)。

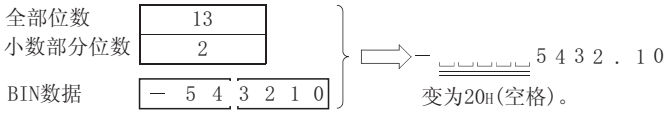


小数部分位数为 0 时，不存储 2EH(.)。

(c) 小数部分位数的值大于 BIN 数据的位数的情况下，将自动地附加 0 并向右对齐而转换为 0.□□□□。



(d) 从全部位数的值中减去符号、小数点部分的位数后仍大于 BIN 数据的位数的情况下，在符号与数值之间将存储 20H(空格)。



BIN 数据的位数一方大的情况下，将变为出错状态。

(e) 在转换后的字符串的最后将自动存储 00H。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

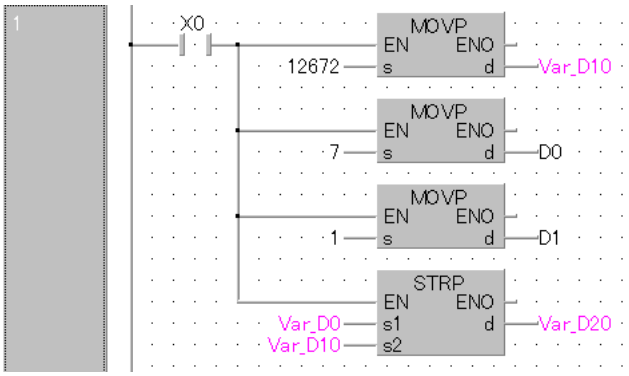
- ㉑ 中指定的全部位数指定超出了下述范围时。 (出错代码： 4100)
  使用 STR(P) 指令时..... 2 ～ 8
  使用 DSTR(P) 指令时..... 2 ～ 13
- ㉑ +1 中指定的小数部分位数指定超出了下述范围时。 (出错代码： 4100)
  使用 STR(P) 指令时..... 0 ～ 5
  使用 DSTR(P) 指令时..... 0 ～ 10
- ㉑ 中指定的全部位数与㉑ +1 中指定的小数部分位数的指定值的关系不符合下述公式时。
  (出错代码： 4100)
  全部位数 -3 ≥小数部分位数
- ㉑ 中指定的位数小于㉒ 中指定的 BIN 数据的位数 +2 时。
  (㉑ 的位数＜不包含㉒ 的符号的 BIN 数据的位数 + 符号 (+ 或者 -) 的位数 + 小数点 (.) 的位数) (出错代码： 4100)
- 存储ⓓ 中指定的字符串的软元件范围超出了相应软元件的范围时。
  (出错代码： 4101)

程序示例

(1) 以下为将 X0 置为 ON 时，将 Var_D10 中存储的 BIN16 位数据按照 D0,D1 的位数指定转换为字符串后，存储到 Var_D20 中的程序。

(Var_D0 被宣言为被分配了 D0、D1 的全局标签。)

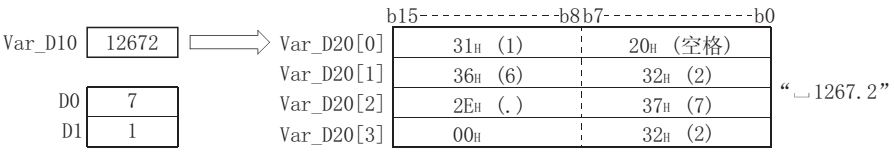
[结构体梯形图]



[ST]

```
IF X0 THEN
    MOVP(TRUE,12672,Var_D10);
    MOVP(TRUE,7,D0);
    MOVP(TRUE,1,D1);
    STRP(TRUE,Var_D0,Var_D10,Var_D20);
END_IF;
```

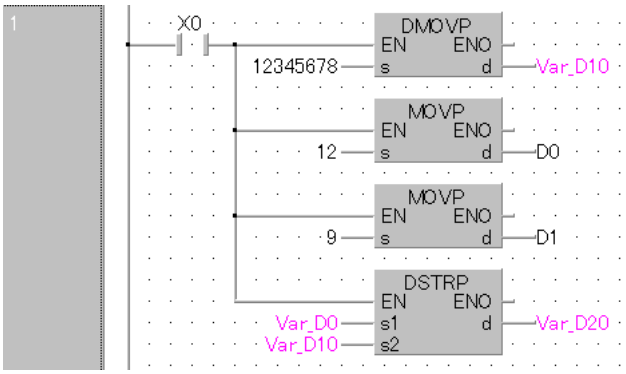
[动作]



(2) 以下为将 X0 置为 ON 时，将 Var_D10 中存储的 BIN32 位数据按照 D0,D1 的位数指定转换为字符串后，存储到 Var_D20 中的程序。

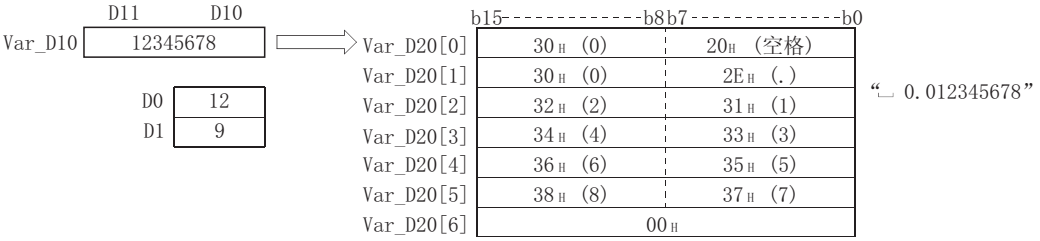
(Var_D0 被宣言为被分配了 D0、D1 的全局标签。)

[结构体梯形图]




```
[ST]
IF X0 THEN
    DMOVP(TRUE,12345678,Var_D10);
    MOVP(TRUE,12,D0);
    MOVP(TRUE,9,D1);
    DSTRP(TRUE,Var_D0,Var_D10,Var_D20);
END_IF;
```

[动作]



7.11.10 字符串→BIN16 位 /32 位转换

VAL, DVAL

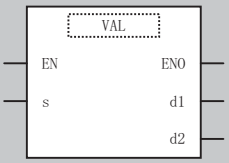
Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

VAL(P)
DVAL(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= VAL (EN, s, d1, d2);

[ ] 中放入下述指令。

VAL	VALP
DVAL	DVALP

输入自变量，EN：　执行条件　：位
　　s：　转换为 BIN 数据的字符串　：ANY16 的数组 (0..4)/(0..6)
　　：字符串 (8)

输出自变量，ENO：　执行结果　：位
　　d1：　转换后的 BIN 数据的位数　：ANY32
　　d2：　换后的 BIN 数据　：ANY16/32

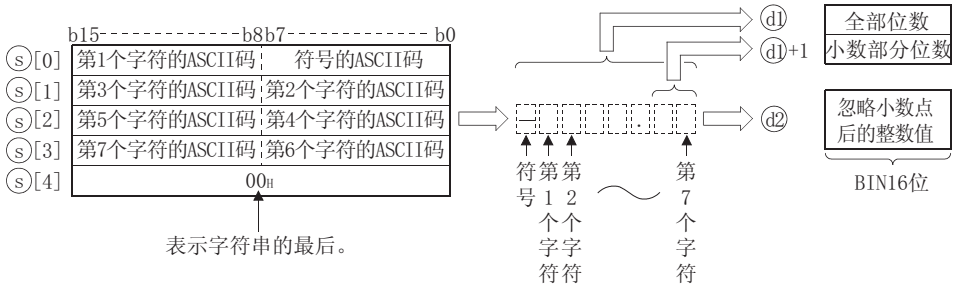
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	－	○		－				○	－
ⓓ1	○	○		－				－	－
ⓓ2	○	○		○				－	－

功 能

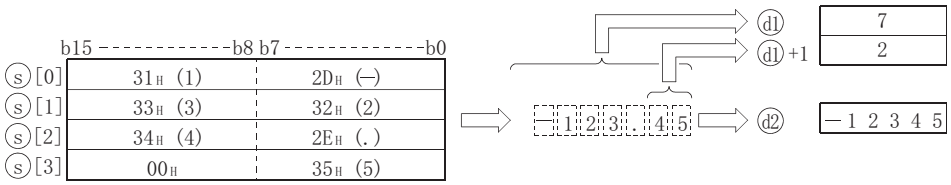
VAL(P)

(1) 将ⓢ中指定的软元件编号以后中存储的字符串转换为 BIN16 位数据后，将位数及 BIN 数据存储到ⓓ1，ⓓ2 中。

在字符串→ BIN 转换中，将从ⓢ中指定的软元件编号开始至存储了 00H 的软元件编号为止的数据作为字符串处理。



例如ⓢ以后指定了 -123.45 的字符串的情况下，将按以下方式存储到ⓓ1，ⓓ2 中。



(2) ⓢ中可指定的字符串的全部字符数为 2 ～ 8 个字符。

(3) 在ⓢ中指定的字符串中，小数部分的字符数为 0 ～ 5 个字符。
但是，应设置为（全部位数 -3）以下。

(4) 可转换为 BIN 值的数值的字符串的范围为，忽略小数点后的值 -32768 ～ 32767。
此外，对于除符号及小数点以外数值的字符串，只能在 30H ～ 39H 的范围内进行指定。
忽略小数点后的值的情况如下所示。

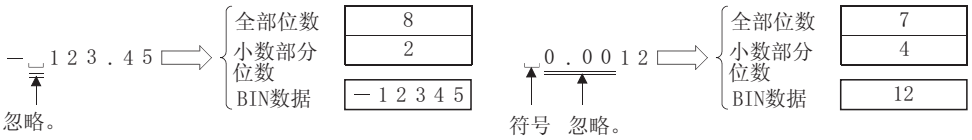
例 -12345.6 → -123456

(5) 在符号中，表示为正数值的情况下设置 20H，表示负数值的情况下设置 2DH。

(6) 在小数点中设置 2EH。

(7) 在ⓓ1中存储的全部位数中，存储表示数值的字符（包含符号、小数点）的所有的字符数。
在ⓓ1+1 中存储的小数部分位数中，存储表示 2EH(.) 以后的小数部分的字符数。
在ⓓ2 中存储的 BIN 数据中，将忽略了小数点后的字符串转换为 BIN 值后进行存储。

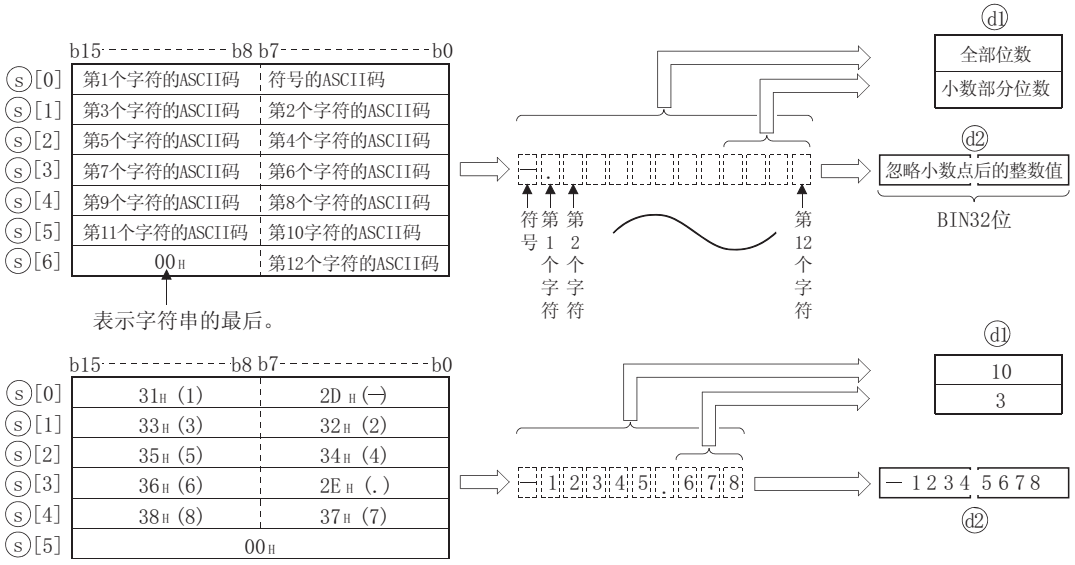
(8) 在ⓢ中指定的字符串中，在符号与最初的除 0 以外的数值之间，存在有 20H（空格）或 30H(0) 的情况下，将在忽略 20H，30H 的状况下转换为 BIN 值。



DVAL(P)

(1) 将ⓢ中指定的软元件编号以后中存储的字符串转换为 BIN32 位数据后，将位数及 BIN 数据存储到ⓓ1, ⓓ2 中。

在字符串→ BIN 转换中，将从ⓢ中指定的软元件编号开始至存储了 00H 的软元件编号为止的数据作为字符串处理。



(2) ⓢ中可指定的字符串的全部字符数为 2 ～ 13 个字符。

(3) 在ⓢ中指定的字符串中，小数部分的字符数为 0 ～ 10 个字符。
但是，应设置为（全部位数 -3）以下。

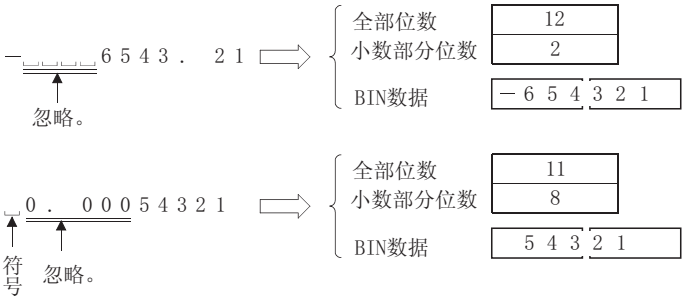
(4) 可转换为 BIN 值的数值的字符串的范围为，忽略小数点后的值 -2147483648 ～ 2147483647。
此外，对于除符号及小数点以外数值的字符串，只能在 30H ～ 39H 的范围内进行指定。

(5) 在符号中，表示为正数值的情况下设置 20H，表示负数值的情况下设置 2DH。

(6) 在小数点中设置 2EH。

(7) 在ⓓ1中存储的全部位数中，存储表示数值的字符（包含符号、小数点）的所有的字符数。
在ⓓ1+1 中存储的小数部分位数中，存储表示 2EH(.) 以后的小数部分的字符数。。
在ⓓ2 中存储的 BIN 数据中，将忽略了小数点后的字符串转换为 BIN 值后进行存储。

(8) 在ⓢ中指定的字符串中，在符号与最初的除 0 以外的数值之间，存在有 20H（空格）或 30H(0) 的情况下，将在忽略 20H, 30H 的状况下转换为 BIN 值。



出 错

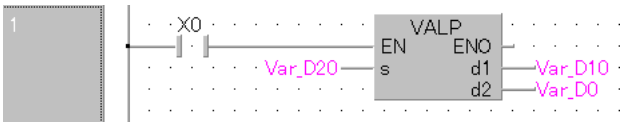
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的字符串的字符数超出了下述范围时。
  （出错代码： 4100）
  使用 VAL(P) 指令时..... 2 ～ 8 个字符
  使用 DVAL(P) 指令时..... 2 ～ 13 个字符
- ⓢ中指定的字符串的小数部字符数超出了下述范围时。
  （出错代码： 4100）
  使用 VAL(P) 指令时..... 0 ～ 5 个字符
  使用 DVAL(P) 指令时..... 0 ～ 10 个字符
- ⓢ中指定的全部字符数与小数部分字符数的关系不符合下述公式时。
  （出错代码： 4100）
  全部字符数 -3 ≥小数部分字符数
- 符号中设置了除 20H, 2DH 以外的 ASCII 码时。
  （出错代码： 4100）
- 各数字的位中设置了除 30H ～ 39H 及 2EH(小数点) 以外的 ASCII 码时。
  （出错代码： 4100）
- 设置了多个小数点时。（出错代码： 4100）
- 转换后的 BIN 值的值超出了下述范围时。（出错代码： 4100）
  使用 VAL(P) 指令时..... -32768 ～ 32767
  使用 DVAL(P) 指令时..... -2147483648 ～ 2147483647
- ⓢ中指定的软元件编号开始至相应软元件的最终软元件编号为止之间未设置 00H 时。
  （出错代码： 4101）

程序示例

(1) 以下为将 X0 置为 ON 时，将 Var_D20 中存储的字符串数据视为整数值转换为 BIN 值后，存储到 Var_D0 中的程序。

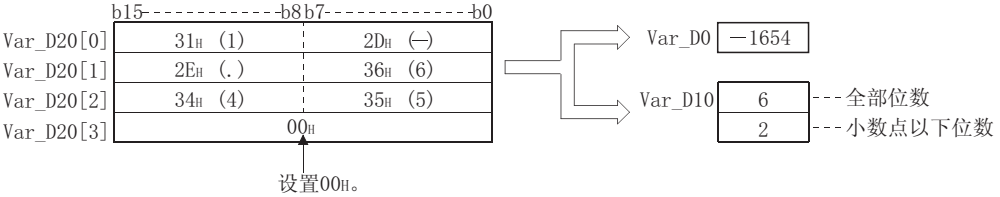
[结构体梯形图]



[ST]

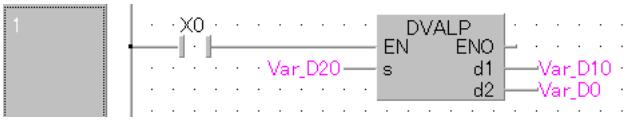
```
VALP(X0,Var_D20,Var_D10,Var_D0);
```

[动作]



(2) 以下为将 X0 置为 ON 时，将 Var_D20 中存储的字符串数据视为整数值转换为 BIN 值后，存储到 Var_D0 中的程序。

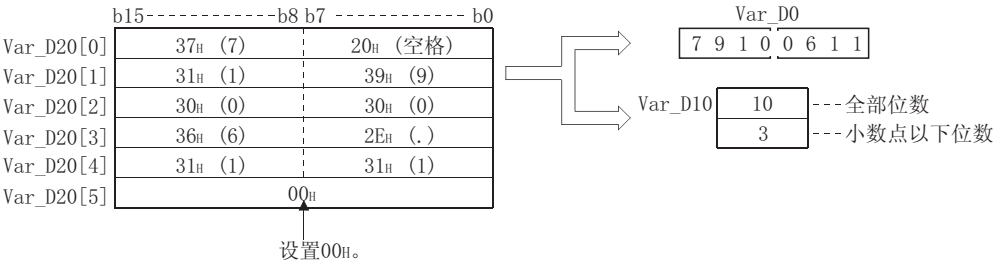
[结构体梯形图]



[ST]

```
DVALP(X0,Var_D20,Var_D10,Var_D0);
```

[动作]



7.11.11 浮动小数点→字符串转换

ESTR

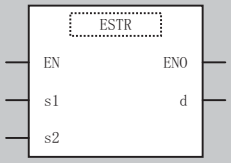
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

ESTR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ESTR (EN, s1, s2, d);

[ ] 中放入下述指令。

ESTR　　ESTRP

输入自变量，	EN:	执行条件	: 位
	s1:	转换的浮动小数点数据	: 单精度实数
	s2:	转换的数值的显示指定	: ANY16 的数组 (0..2)
输出自变量，	ENO:	执行结果	: 位
	d:	转换后的字符串	: 字符串

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ1	–	○		–	○		–	○	–
ⓢ2	–	○		–	–		–	–	–
ⓓ	–	○		–	–		–	–	–

功 能

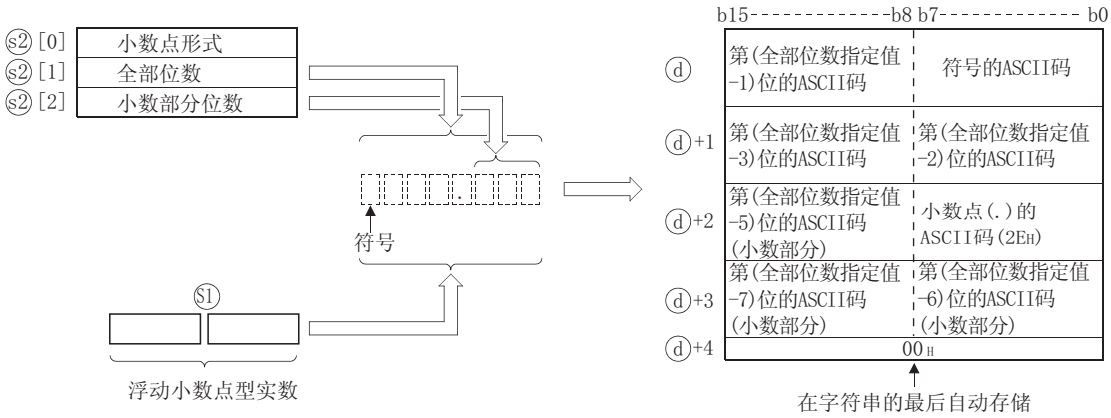
(1) 将ⓢ1中指定的浮动小数点型实数数据，按照ⓢ2中指定的显示指定转换为字符串后，存储到ⓓ中指定的软元件编号以后。

ⓢ1中可指定的范围为 0 或± 2^{-126} ～± 2^{128}。

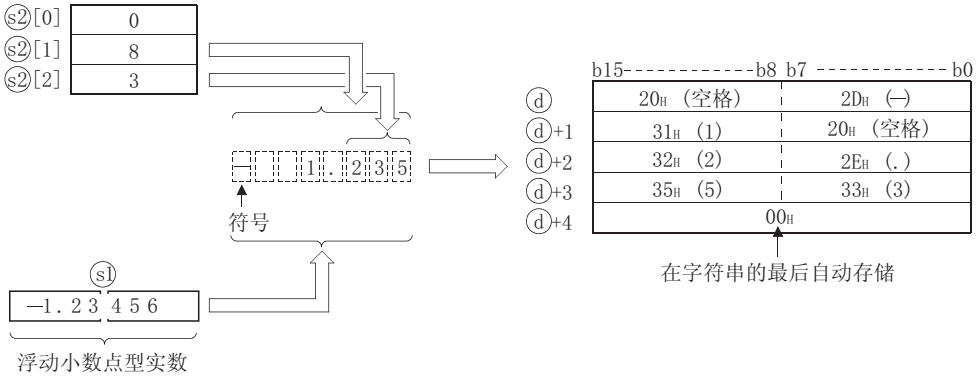
(2) 根据ⓢ2中指定的显示指定情况转换后的数据有所不同。

ⓢ2[0]	0：小数点格式 1：指数格式	根据ⓢ2的格式转换后的数据有所不同。
ⓢ2[1]	全部位数	可在2～24的范围内进行设置。
ⓢ2[2]	小数部分位数	

小数点格式的情况



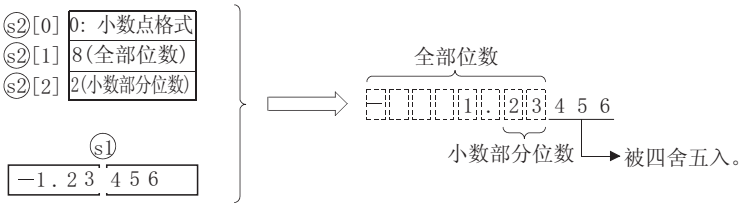
例如全部位数为 8，小数部分位数为 3，指定了 -1.23456 的情况下，按以下方式存储到ⓓ以后。



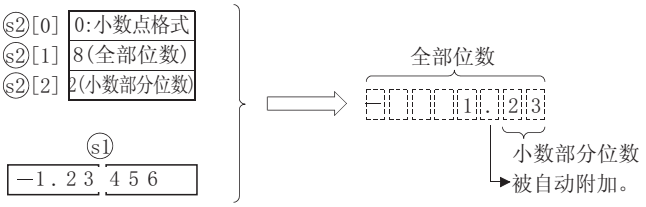
(a) ⓢ2 +1 中可指定的全部位数如下所示。
小数部分位数为 0 时 ..... 位数（最大： 24)≧2
小数部分位数为 0 以外时 ..... 位数（最大： 24)≧(小数部分位数 +3)

(b) ⓢ2 +2 中可指定的小数部分位数为 0 ～ 7 位。
但是，应设置为小数部分位数≤（全部位数 -3)。

(c) 转换后的字符串数据按如下所示存储到ⓓ以后的软元件编号中。
1) 在整数部分的符号中，浮动小数点型实数数据为正时存储 20H(空格)，为负时存储 2DH(-)。
2) 在小数部分位数的范围中，容纳不下浮动小数点型实数数据的小数部分的情况下，将低位的小数部分进行四舍五入。

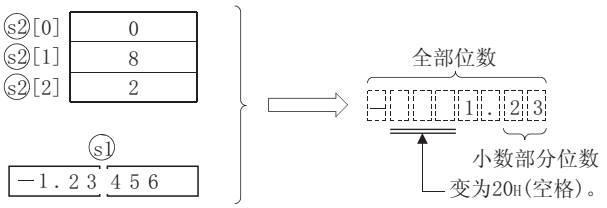


3) 将小数部分位数设置为 0 以外的情况下，在指定的小数部分位数 +1 位中将自动地存储 2EH(.)。



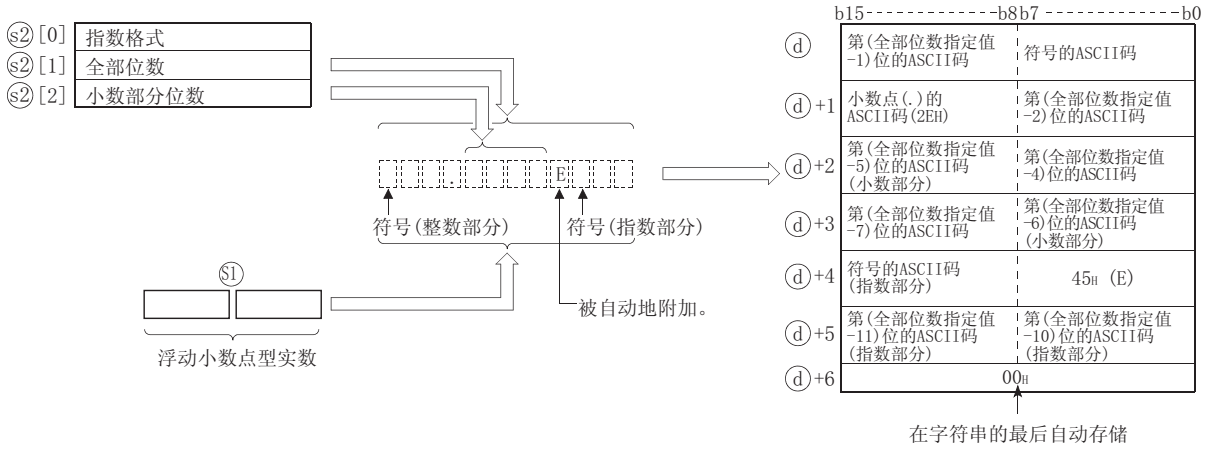
小数部分位数为 0 时，不存储 2EH(.)。

4) 从全部位数中减去符号、小数点、小数部分的位数后仍大于浮动小数点型实数数据的整数部分的情况下，在符号与整数部分之间存储 20H(空格)。

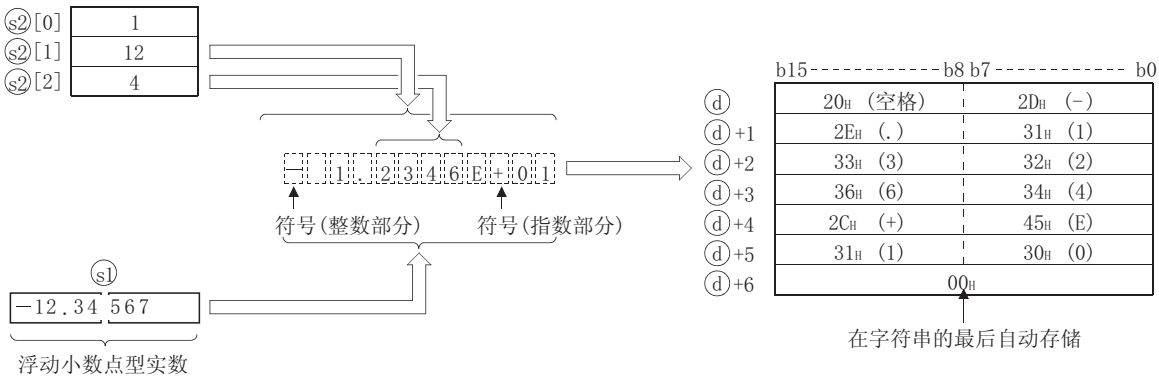


5) 在转换后的字符串的最后将自动地存储 00H。

指数格式的情况



例如全部位数为 12，小数部分位数为 4，指定了 -12.34567 的情况下，按以下方式存储到 ⓓ 以后。



(a) ㉒+1 中可指定的全部位数如下所示。

小数部分位数为 0 时 ..... 位数（最大： 24)≧2

小数部分位数为 0 以外时 ..... 位数（最大： 24)≧（小数部分位数 +7)

(b) ㉒+2 中可指定的小数部分位数为 0 ～ 7 位。

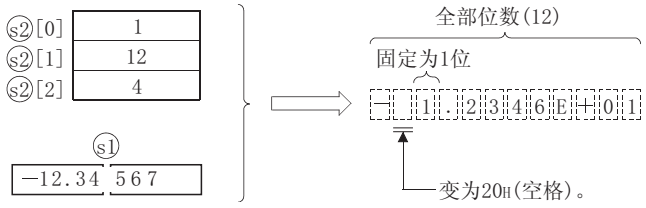
但是，应设置为小数部分位数≤（全部位数 -7)。

(c) 转换后的字符串数据按如下所示存储到㉔以后的软元件编号中。

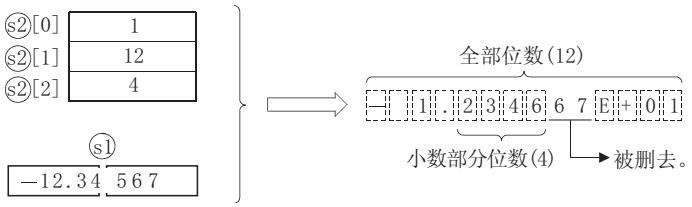
1) 在整数部分的符号中，浮动小数点型实数数据为正时存储 20H（空格），为负时存储 2DH(-)。

2) 整数部分固定为 1 位。

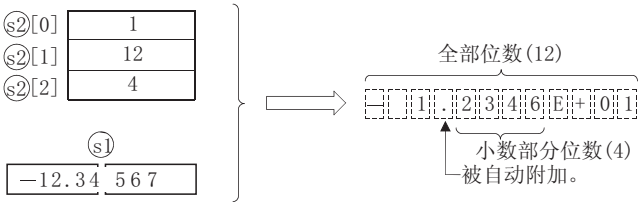
整数部分与符号之间存储 20H（空格）。



3) 在小数部分位数的范围中，容纳不下浮动小数点型实数数据的小数部分的情况下，将低位的小数部分进行四舍五入。



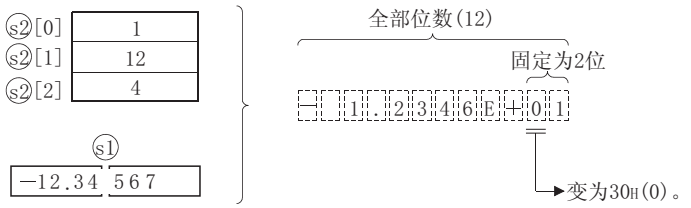
4) 将小数部分位数设置为 0 以外的情况下，在指定的小数部分位数 +1 位中将自动地存储 2EH(.)。



小数部分位数为 0 时，不存储 2EH(.)。

5) 在指数部分的符号中，指数为正时存储 2CH(+)，为负时存储 2DH(-)。

6) 指数部分固定为 2 位。
指数部分为 1 位的情况下，在与指数部分的符号之间存储 30H(0)。



7) 在转换后的字符串的最后将自动地存储 00H。

出 错

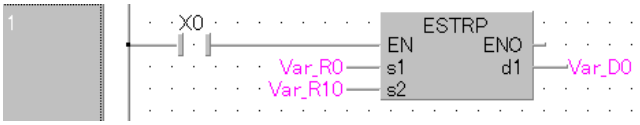
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- (s1) 超出了以下范围时。 (出错代码： 4100)
  0, $\pm 2^{-126} \leqq$ (s1) $< \pm 2^{128}$
- (s2) 中指定的格式指定为 0、1 以外时。 (出错代码： 4100)
- (s2)+1 中指定的全部位数指定超出了下述范围时。 (出错代码： 4100)
  小数点格式的情况下
  小数部分位数为 0 时..... 全部位数≧2
  小数部分位数为 0 以外时..... 全部位数≧(小数部分位数 +3)
  指数格式的情况下
  小数部分位数为 0 时..... 全部位数≧6
  小数部分位数为 0 以外时..... 全部位数≧(小数部分位数 +7)
- (s2)+2 中指定的小数部分位数指定超出了下述范围时。 (出错代码： 4100)
  小数点格式的情况下..... 小数部分位数≦(全部位数 -3)
  指数格式的情况下..... 小数部分位数≦(全部位数 -7)
- 存储 (d) 中指定的字符串的软元件范围超出了相应软元件的范围时。
  (出错代码： 4101)
- (s2) 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  (通用型 QCPU、LCPU 时) (出错代码： 4140)

程序示例

(1) 以下为 X0 变为 ON 时，将 Var_R0 中存储的浮动小数点型实数数据，按照 Var_R10 中存储的转换指定进行转换后，存储到 Var_D0 以后的程序。

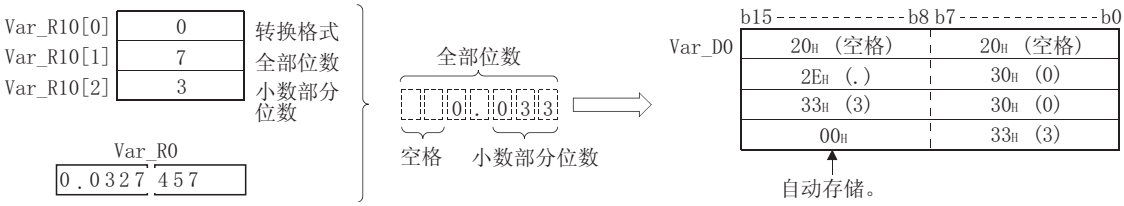
[结构体梯形图]



[ST]

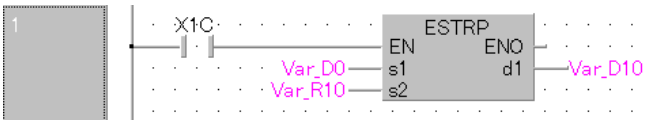
```
ESTRP(X0,Var_R0,Var_R10,Var_D0);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将 Var_D0 中存储的浮动小数点型实数数据，按照 Var_R10 中存储的转换指定进行转换后，存储到 Var_D10 以后的程序。

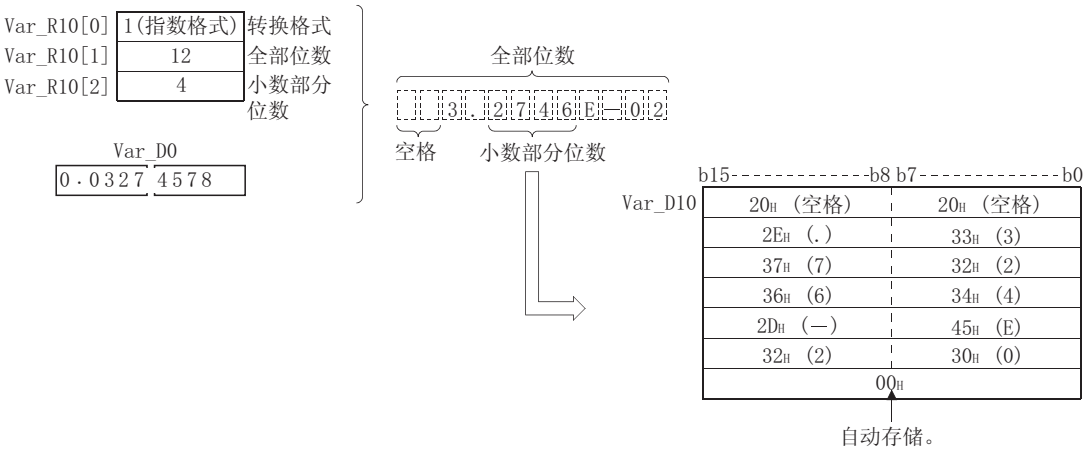
[结构体梯形图]



[ST]

```
ESTRP(X1C,Var_D0,Var_R10,Var_D10);
```

[动作]



7.11.12 字符串→浮动小数点转换

EVAL

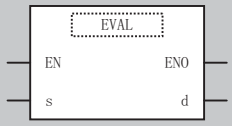
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

EVAL(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EVAL (EN, s, d);

□ 中放入下述指令。

EVAL　　EVALP

输入自变量，EN:　执行条件　：位

　　s:　转换为浮动小数点型实数数据的字符串数据　：字符串 (24)

输出自变量，ENO:　执行结果　：位

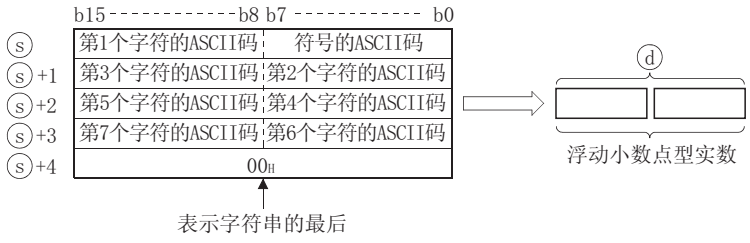
　　d:　转换后的浮动小数点型实数数据　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ	－	○		－		－	－	○	－
ⓓ	－	○		－		○	－	－	－

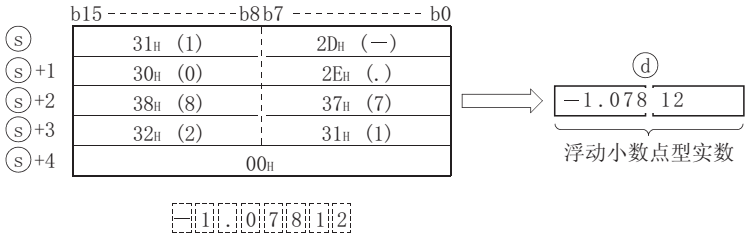
功 能

(1) 将ⓢ中指定的软元件编号以后中存储的字符串转换为浮动小数点型实数后，存储到ⓓ中指定的软元件中。

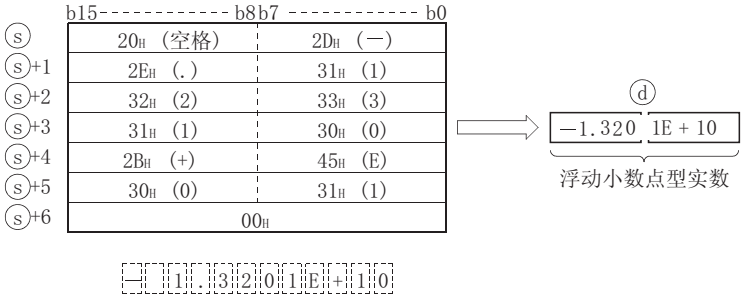
(2) 指定的字符串无论是小数点格式还是指数格式均可转换为浮动小数点型实数数据。



(a) 小数点格式的情况

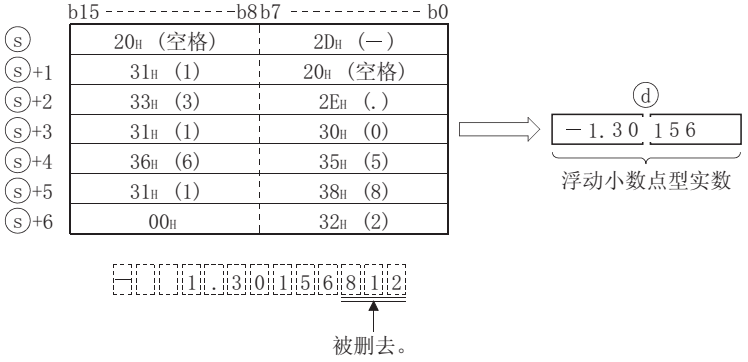


(b) 指数格式的情况

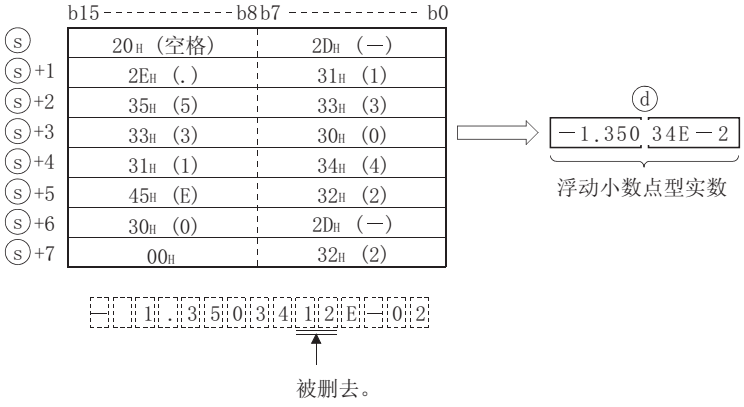


(3) 在ⓢ中指定的字符串中，转换为浮动小数点型实数的字符串的除符号、小数点、指数部分以外的 6 位数有效，第 7 位数以后在转换时将被删除。

(a) 小数点格式的情况



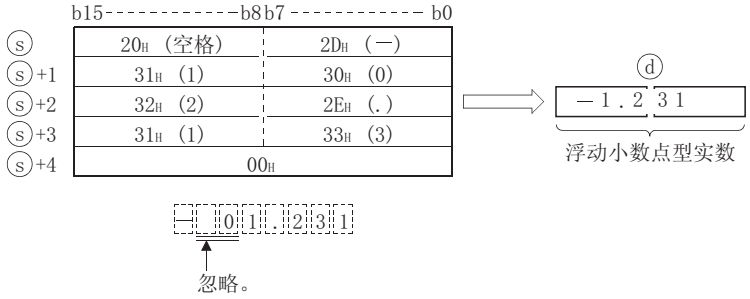
(b) 指数格式的情况



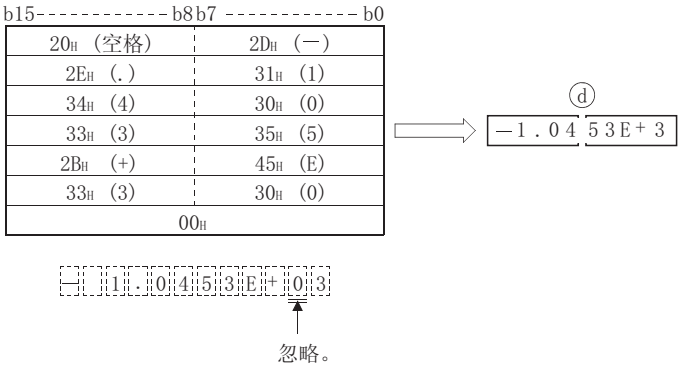
(4) 在小数点格式中如果在符号中指定 2BH(+) 或将符号省略，将被作为正的值进行转换。
此外，如果在符号中指定 2DH(−)，将被作为负的值进行转换。

(5) 在指数格式中如果在指数部分的符号中指定 2BH(+) 或将符号省略，将被作为正的值进行转换。
如果在指数部分的符号中指定 2DH(−)，将被作为负的值进行转换。

(6) 在ⓢ中指定的字符串中，在符号与最初的除 0 以外的数值之间，存在有 20H(空格) 或 30H(0) 的情况下，将在忽略 20H, 30H 的状况下进行转换。



(7) 在指数格式的字符串中，在 E 与数值之间存在 30H(0) 的情况下，将在忽略 30H 的状况下进行转换。



(8) 在字符串中包含有 20H(空格) 的情况下，将在忽略 20H 的状况下进行转换。

(9) 字符串最多可设置 24 个字符。
字符串中的 20H(空格), 30H(0) 也作为 1 个字符计数。

出 错

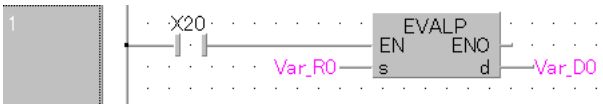
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 整数部分、小数部分中有除 30H(0) ～ 39H(9) 以外的字符时。 (出错代码： 4100)
- ⓓ中指定的字符串的中有 2 个及以上的 2EH(.) 时。 (出错代码： 4100)
- 指数部分中存在除 45H(E), 2BH(+), 45H(E), 2DH(-) 以外的字符时，或存在多个指数部分时。 (出错代码： 4100)
- 转换后的数据超出以下范围时。 (出错代码： 4100)
- 0, $2^{-126} \leqq |$ 转换后的数据 $| < 2^{128}$
- ⓢ算起的相应软元件范围内没有 00H 时。 (出错代码： 4101)
- ⓢ以后的字符数为 0 或超过 24 个字符时。 (出错代码： 4100)

程序示例

(1) 以下为 X20 变为 ON 时，将 Var_R0 以后存储的字符串转换为浮动小数点型实数后，存储到 Var_D0 中的程序。

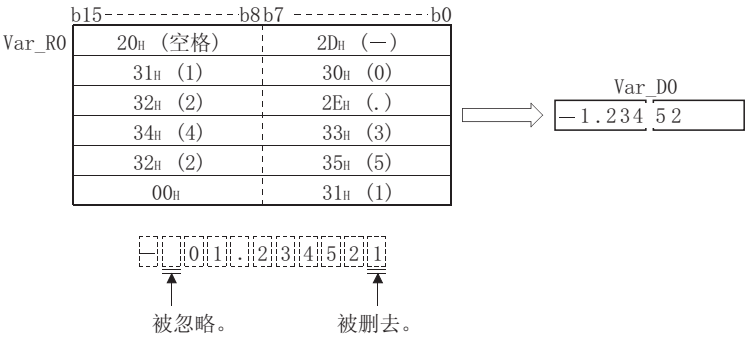
[结构体梯形图]



[ST]

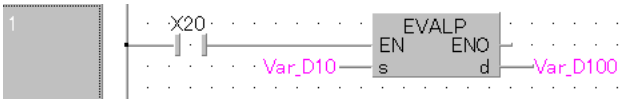
```
EVALP(X20,Var_R0,Var_D0);
```

[动作]



(2) 以下为 X20 变为 ON 时，将 Var_D10 以后存储的字符串转换为浮动小数点型实数后，存储到 Var_D100 中的程序。

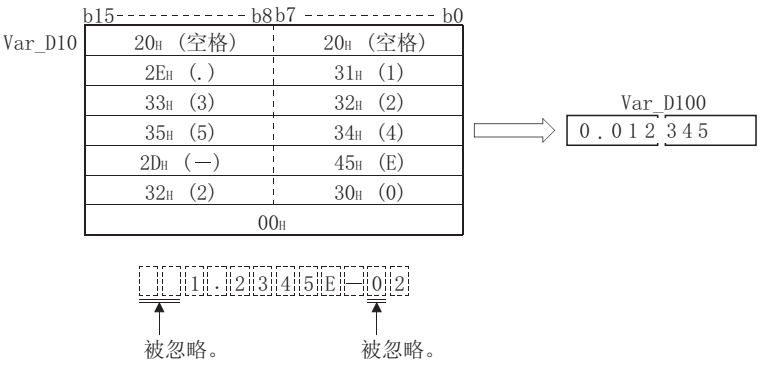
[结构体梯形图]



[ST]

```
EVALP(X20,Var_D10,Var_D100);
```

[动作]



7.11.13 16 进制 BIN → ASCII 转换

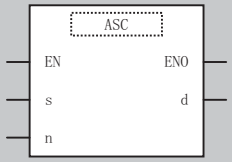
ASC

Basic | High performance | Universal | L CPU

ASC(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ASC (EN, s, n, d);

[　　] 中放入下述指令。

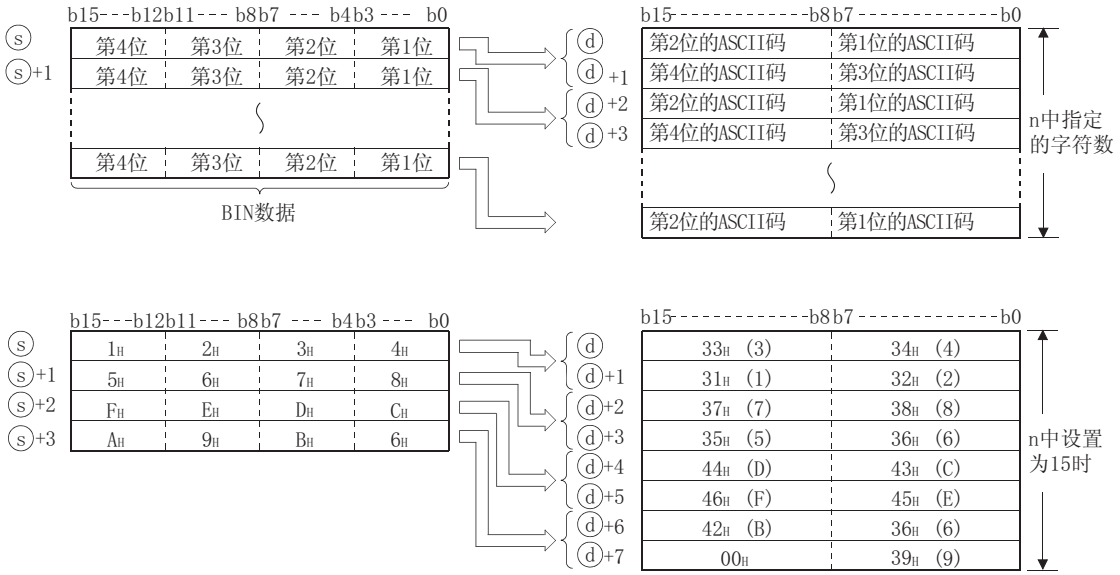
ASC　　ASCP

输入自变量，EN:　执行条件　：位
　　s:　存储转换为字符串的 BIN 数据的软元件的起始编号　：ANY16
　　n:　存储的字符数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　转换后的字符串　：ANY16
　　：字符串

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	－	○		－					－
n	－	○			○				－
ⓓ	－	○		－					－

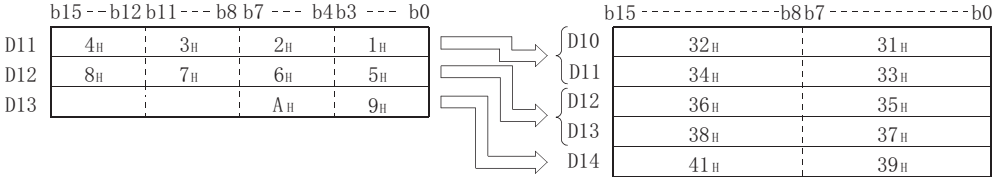
功 能

(1) 将ⓢ中指定的软元件编号以后中存储的 BIN16 位数据，通过 16 进制数处理转换为 ASCII。然后，以 n 中指定的字符数的范围存储到ⓓ中指定的软元件编号以后。



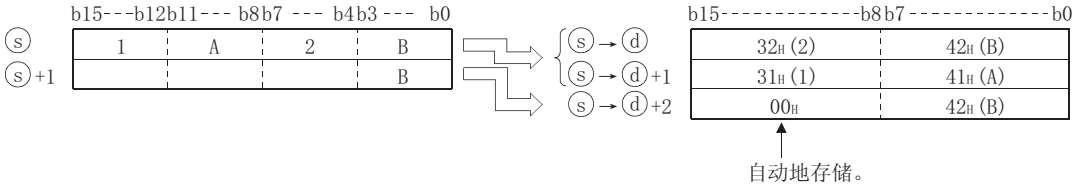
(2) 通过在 n 中设置字符数，ⓢ 中指定的 BIN 数据的范围以及ⓓ 中指定的字符串的存储软元件的范围将自动确定。

(3) 存储转换的 BIN 数据的软元件范围与存储转换后的 ASCII 数据的软元件范围相重叠的情况下，也可正常处理。



(4) n 中指定的字符数为奇数的情况下，在存储字符串的软元件范围的最终软元件编号的高 8 位中，将自动地存储 00H。

n 的字符数为 5 的情况下



(5) n 中指定的字符数为 “0” 的情况下，不进行转换处理。

出 错

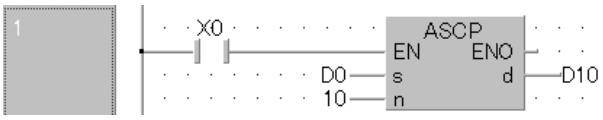
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的软元件编号以后，n 中指定的字符数的范围超出了相应软元件的范围时。
  (出错代码： 4101)
- ⓓ 中指定的软元件编号以后，n 中指定的字符数的范围超出了相应软元件的范围时。
  (出错代码： 4101)

程序示例

以下为将 X0 置为 ON 时，将 D0 中存储的 BIN 数据以 16 进制数转换为字符串后，存储到 D10 ～ D14 中的程序。

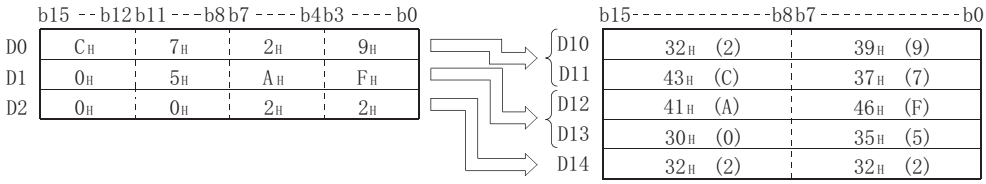
[结构体梯形图]



[ST]

```
ASCP(X0,D0,10,D10);
```

[动作]



7.11.14 ASCII → 16 进制数据 BIN 转换

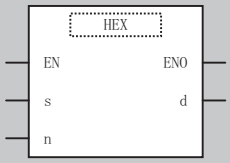
HEX

Basic | High performance | Universal | L CPU

HEX(P)

P: 执行条件　　: ↑

结构体梯形图



ST

ENO:= HEX (EN, s, n, d);

□ 中放入下述指令。

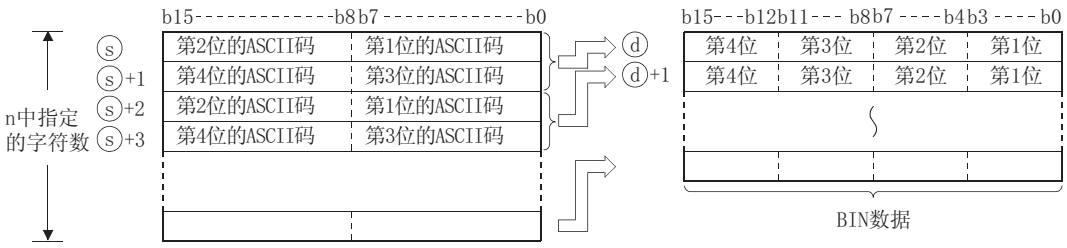
HEX　　HEXP

输入自变量，EN:　执行条件　: 位
　　s:　转换为 BIN 数据的字符串　: ANY16 字符串
　　n:　存储的字符数　: ANY16
输出自变量，ENO:　执行结果　: 位
　　d:　存储转换后的 BIN 数据的软元件的起始编号　: ANY16

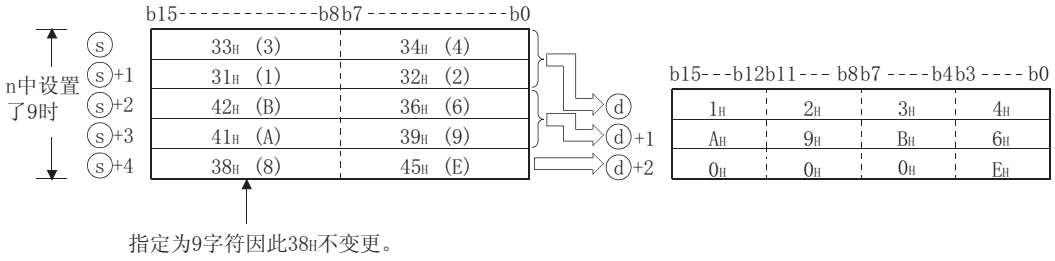
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	-	○		-					-
ⓝ	-	○		○					-
ⓓ	-	○		-					-

功 能

(1) 将ⓢ中指定的软元件编号以后，n 中指定的字符数中存储的 16 进制 ASCII 数据转换为 BIN 值后，存储到ⓓ中指定的软元件编号以后。

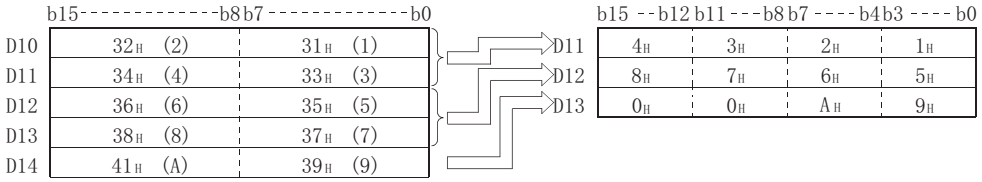


例如 n 中指定了 9 的情况如下所示。

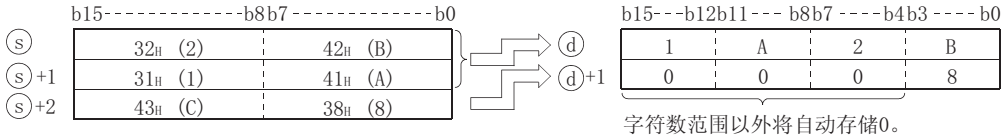


(2) n 中设置了字符数时，ⓢ 中指定的字符串的范围以及存储ⓓ 中指定的 BIN 数据的软元件的范围将自动确定。

(3) 存储要转换的 ASCII 数据的软元件范围与存储转换后的 BIN 数据的软元件范围相重叠的情况下，也可正常处理。



(4) n 中指定的字符数不为 4 的倍数的情况下，在存储转换后的 BIN 值的软元件编号内，在最终软元件编号的指定字符数以后的位数中将自动存储 0。



(5) n 中指定的字符数为 0 的情况下，不进行转换处理。

(6) ⓢ 中可指定的 ASCII 码的范围为 30H ～ 39H, 41H ～ 46H。

出 错

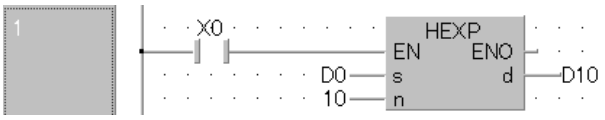
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中设置了 16 进制数值的字符串以外的字符 (30H ～ 39H, 41H ～ 46H 以外的 ASCII 码) 时。 (出错代码 : 4100)
- ⓢ 中指定的软元件编号以后，n 中指定的字符数的范围超出了相应软元件的范围时。 (出错代码 : 4101)
- ⓓ 中指定的软元件编号以后，n 中指定的字符数的范围超出了相应软元件的范围时。 (出错代码 : 4101)
- n 中指定的字符数为负数时。 (出错代码 : 4101)

程序示例

以下为将 X0 置为 ON 时，将 D0 ～ D4 中存储的字符串数据转换为 BIN 数据后，存储到 D10 ～ D12 中的程序。

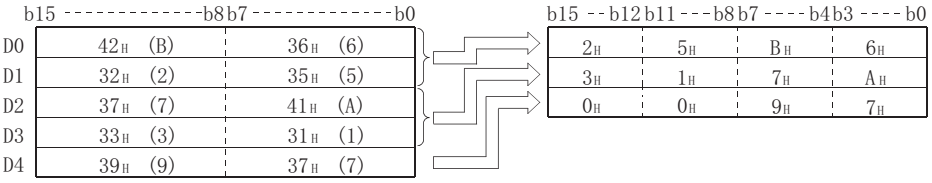
[结构体梯形图]



[ST]

```
HEXP(X0,D0,10,D10);
```

[动作]



7.11.15 从字符串的右侧、左侧的截取

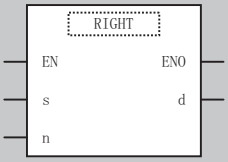
RIGHT, LEFT

Basic | High performance | Universal | L CPU

RIGHT(P)
LEFT(P)

P: 执行条件　　: ↑

结构体梯形图



ST

ENO:= RIGHT (EN, s, n, d);

[　　] 中放入下述指令。

RIGHT	RIGHTP
LEFT	LEFTP

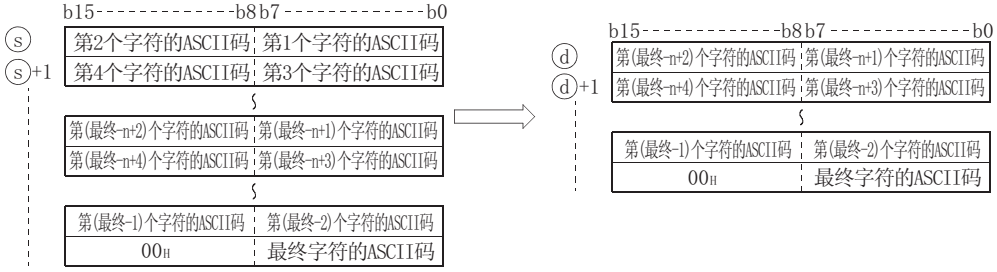
输入自变量，EN:　执行条件　：位
　　s:　字符串　：字符串
　　n:　截取的字符数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　s 的右或左算起 n 字符的字符串　：字符串

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K, H	$	
ⓢ	－	○		－				－	○	－
n	○	○		○				○	－	－
ⓓ	－	○		－				－	－	－

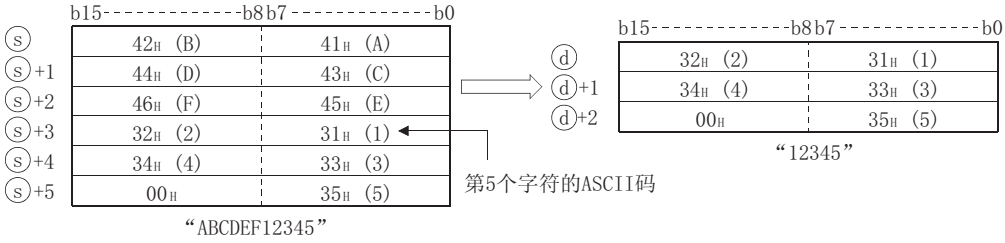
功 能

RIGHT(P)

(1) 将ⓢ中指定的软元件编号以后中存储的字符串数据的右端（字符串的最终）算起的 n 字符的数据，存储到ⓓ中指定的软元件编号以后。



n=5 时

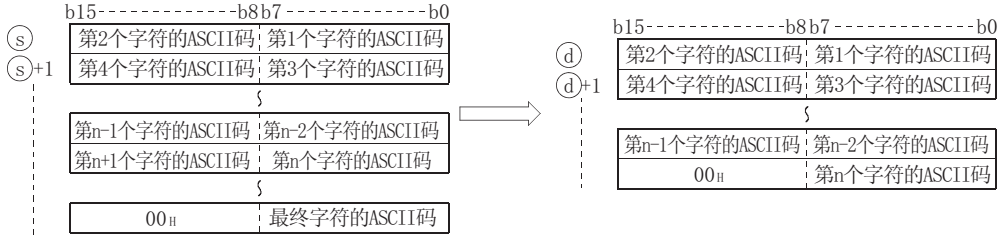


(2) 表示字符串的最终的 NULL 码 (00H) 将被自动地附加在字符串的最后。
关于字符串数据的格式，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

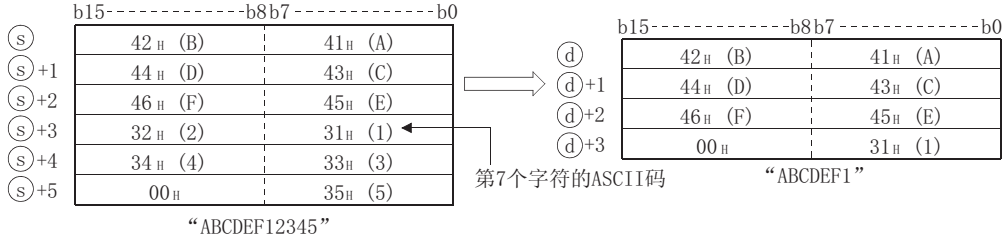
(3) n 中指定的字符数为 0 的情况下，ⓓ 中将被存储 NULL 码 (00H)。

LEFT(P)

(1) 将ⓢ中指定的软元件编号以后中存储的字符串数据的左端（字符串的起始）算起的 n 字符的数据，存储到ⓓ中指定的软元件编号以后。



n=7 时



(2) 表示字符串的最终的 NULL 码 (00H) 将被自动地附加在字符串的最后。
关于字符串数据的格式，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

(3) n 中指定的字符数为 0 的情况下，ⓓ 中将被存储 NULL 码 (00H)。

出 错

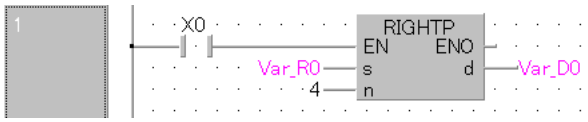
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 超过了ⓢ中指定的字符数时。 （出错代码： 4101）
- ⓓ算起的 n 字符的范围超出了相应软元件的范围时。 （出错代码： 4101）

程序示例

(1) 以下为 X0 变为 ON 时，从 Var_R0 中将存储的字符串数据的右端算起的 4 个字符的数据存储到 Var_D0 以后的程序。

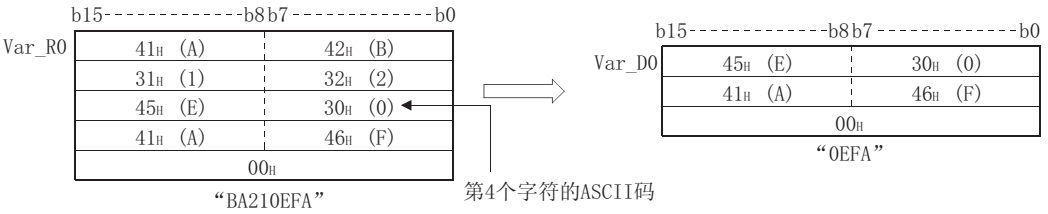
[结构体梯形图]



[ST]

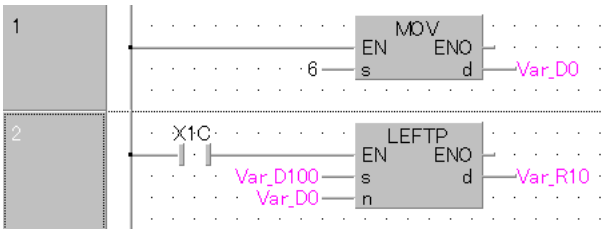
```
RIGHTP(X0,Var_R0,4,Var_D0);
```

[动作]



(2) 以下为 X1C 变为 ON 时，从 Var_D100 中将存储的字符串数据的左端算起的、Var_D0 中存储的值的字符的数据，存储到 Var_R10 以后的程序。

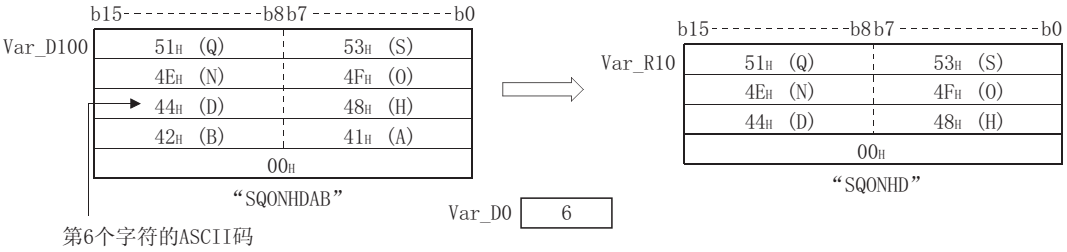
[结构体梯形图]



[ST]

```
Var_D0:=6;
LEFTP(X1C,Var_D100,Var_D0,Var_R10);
```

[动作]



7.11.16 字符串中的任意截取、替换

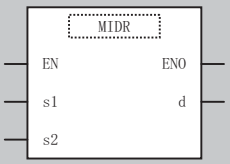
MIDR, MIDW

Basic | High performance | Universal | L CPU

MIDR(P)
MIDW(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= MIDR (EN, s1, s2, d);

[ ] 中放入下述指令。

MIDR	MIDRP
MIDW	MIDWP

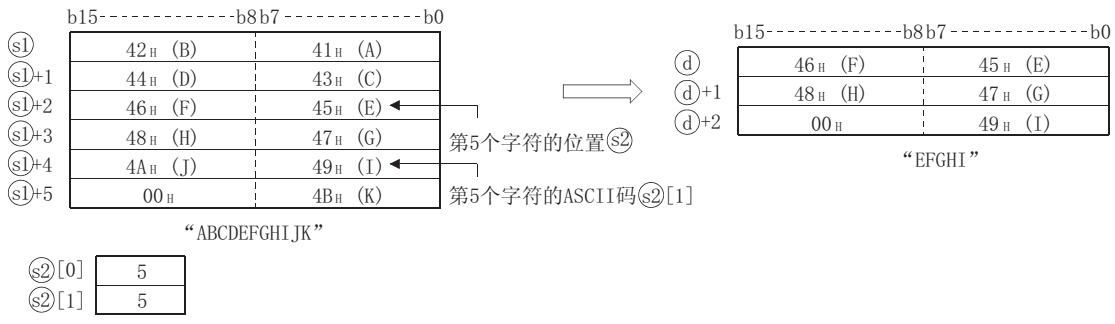
输入自变量， EN: 执行条件 : 位
s1: 字符串 : 字符串
s2: 起始字符的位置及字符数 : ANY16 的数组 (0..1)
s2[0]: 起始字符的位置
s2[1]: 字符数
输出自变量， ENO: 执行结果 : 位
d: 运算结果的字符串数据 : 字符串

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
ⓢ1	–	○		–				○	–
ⓢ2	○	○		○				–	–
ⓓ	–	○		–				–	–

功 能

MIDR(P)

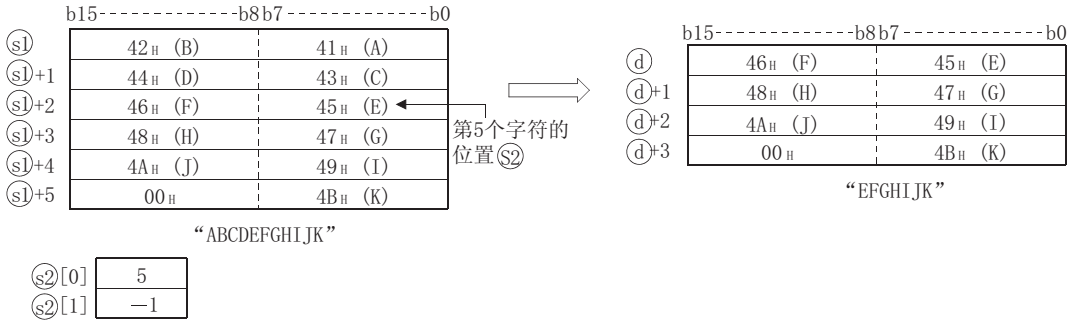
(1) 从ⓢ1中指定的字符串数据的左端开始，将ⓢ2[0]中指定的位置算起的ⓢ2[1]中指定的字符的数据，存储到ⓓ中指定的软元件编号以后。



(2) 表示字符串的最终的 NULL 码 (00H) 将被自动地附加在字符串的最后。
关于字符串数据的格式，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

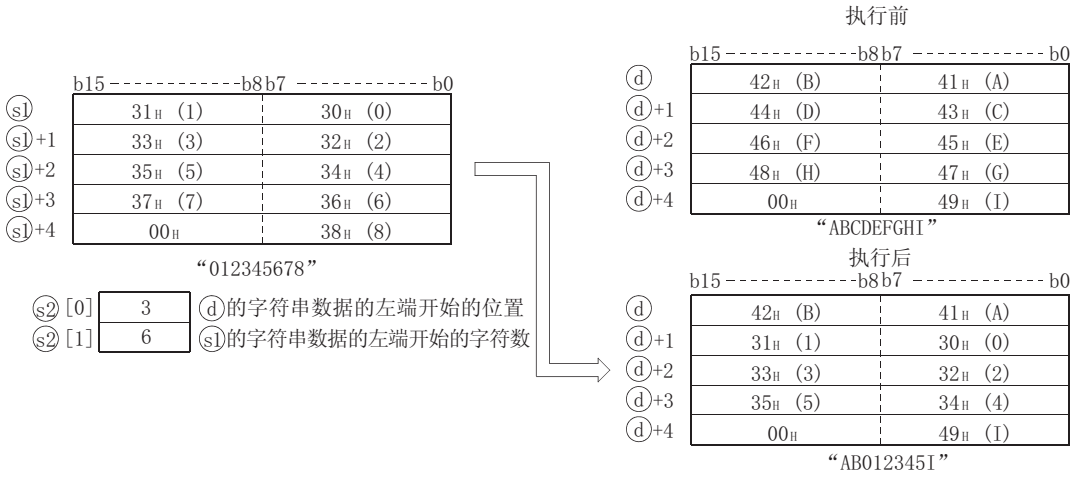
(3) ⓢ2 [1] 中指定的字符数为 0 的情况下，在ⓓ的起始处将存储 NULL 码 (00H)。

(4) ⓢ2 [1] 中指定的字符数为 -1 的情况下，将ⓢ1 中指定的最终字符数据之前的数据，存储到ⓓ中指定的软元件以后。



MIDW(P)

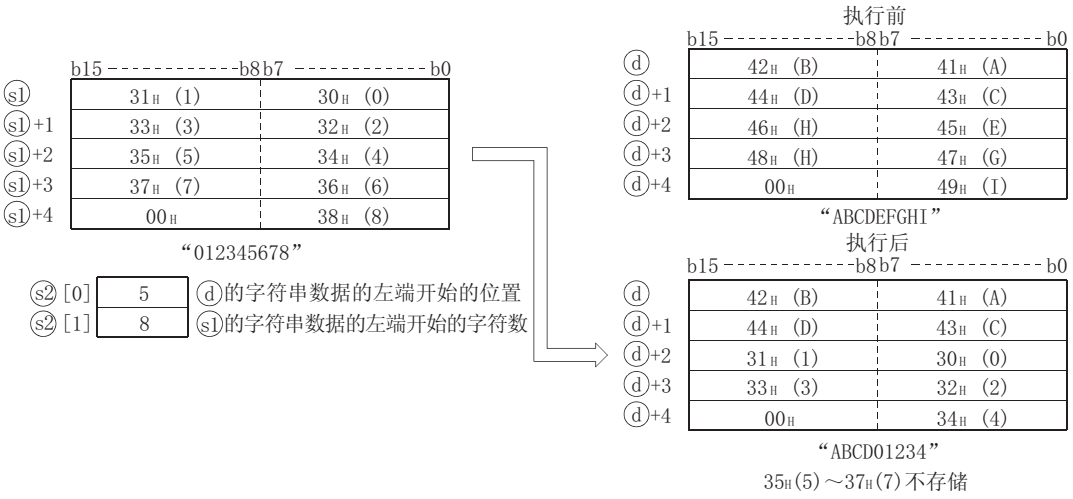
(1) 从ⓢ1 中指定的字符串数据的左端开始，将ⓢ2 [1] 中指定的字符的数据，存储到ⓓ中指定的字符串数据的左端算起的ⓢ2 [0] 中指定的位置以后。



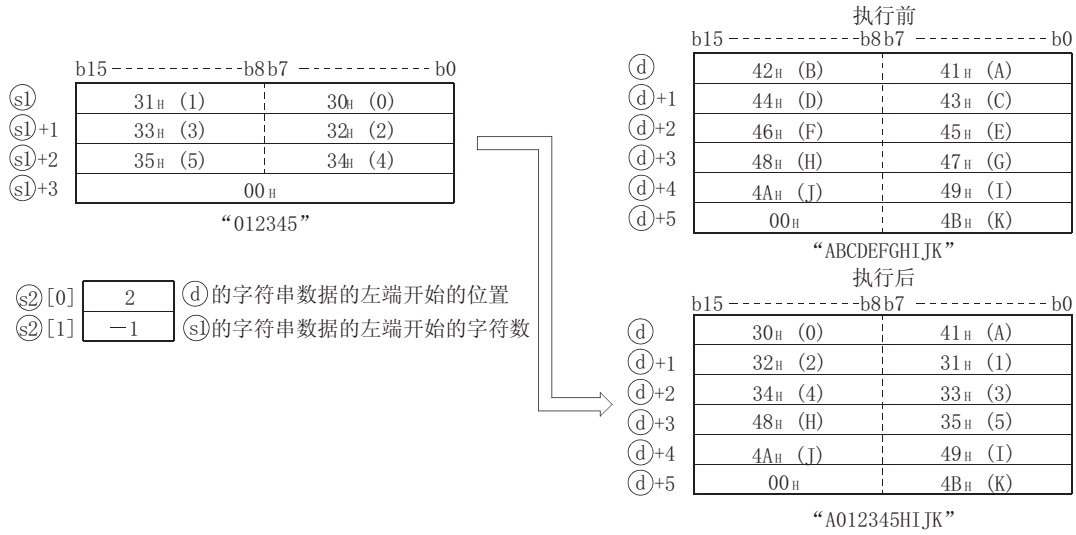
(2) 表示字符串的最终的 NULL 码 (00H) 将被自动地附加在字符串的最后。
关于字符串数据的格式，请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

(3) ⓢ2 [1] 中指定的字符数为 0 的情况下将不进行处理。

(4) ⓢ2[1] 中指定的字符数超过了ⓓ中指定的字符串数据的最终字符的情况下，存储第最终字符之前的数据。



(5) ⓢ2[1] 中指定的字符数为 -1 的情况下，将ⓢ1中指定的最终字符数据之前的数据，存储到ⓓ中指定的软元件以后。



出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

MIDR(P) 指令时

- ⓢ2[0] 的值超过了ⓢ1的字符数时。 (出错代码： 4101)
- ⓓ位置算起的ⓢ2[1] 的字符数超出了ⓓ的软元件范围时。 (出错代码： 4101)

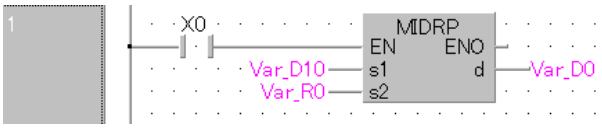
MIDW(P) 指令时

- ⓢ2[0] 的值超过了ⓓ的字符数时。 (出错代码： 4101)
- ⓢ2[1] 的值超过了ⓢ1的字符数时。 (出错代码： 4101)

程序示例

(1) 以下为 X0 变为 ON 时，将 Var_D10 以后存储的字符串数据的左端开始第 3 个字符起至第 6 个字符为止的数据，存储到 Var_D0 以后的程序。

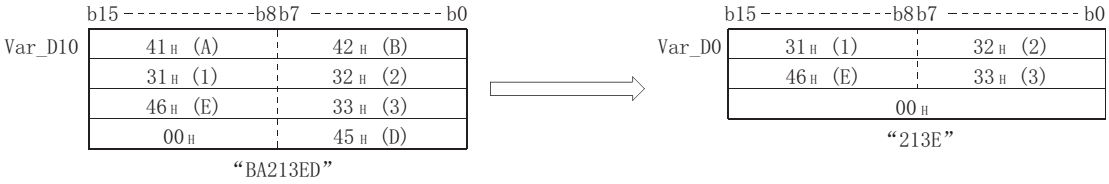
[结构体梯形图]



[ST]

```
MIDRP(X0,Var_D10,Var_R0,Var_D0);
```

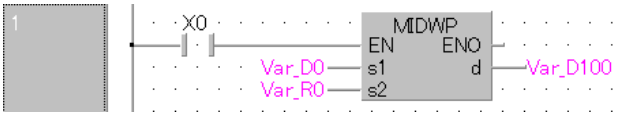
[动作]



Var_R0[0]	3
Var_R0[1]	4

(2) 以下为 X0 变为 ON 时，将 Var_D0 以后存储的字符串数据的 4 个字符，存储到 Var_D100 以后的字符串数据的左端起第 3 个以后的程序。

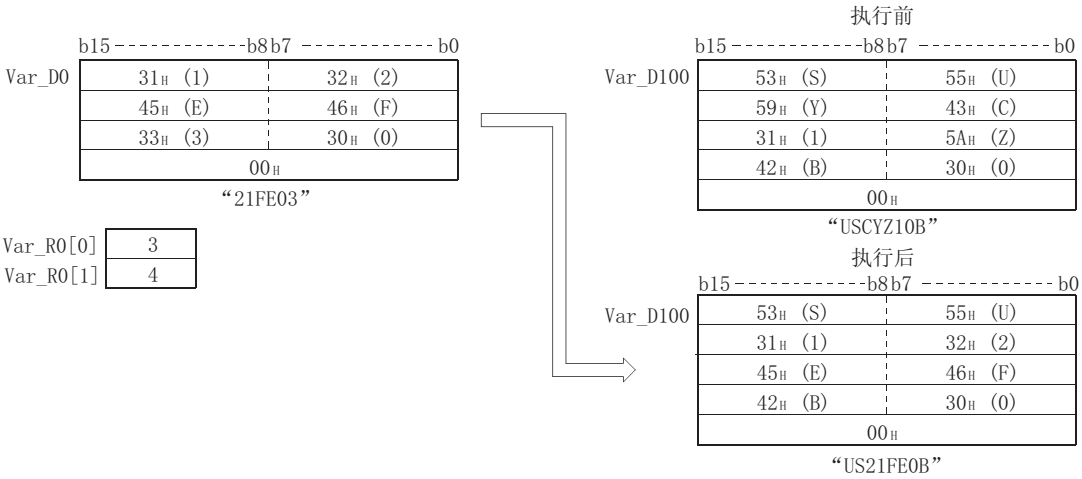
[结构体梯形图]



[ST]

```
MIDWP(X0,Var_D0,Var_R0,Var_D100);
```

[动作]



Var_R0[0]	3
Var_R0[1]	4

7.11.17 字符串查找

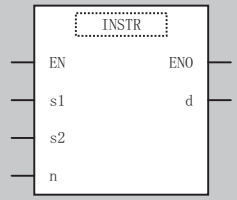
INSTR

Basic | High performance | Universal | L CPU

INSTR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= INSTR (EN, s1, s2, n, d);

□ 中放入下述指令。

INSTR　　INSTRP

输入自变量，EN：　执行条件　：位
　　s1：　查找字符串　：字符串
　　s2：　被查找的字符串　：字符串
　　n：　查找开始位置　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储查找结果的软元件的起始编号　：ANY16

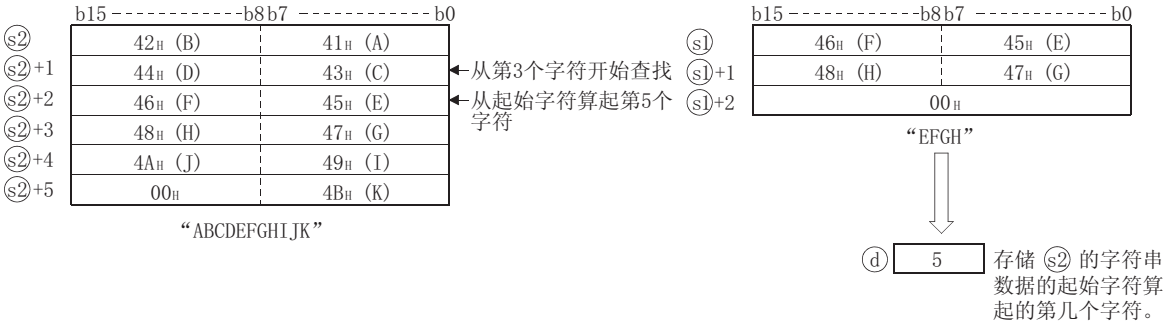
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K, H	$	
⑤1	－	○		－				－	○	－
⑤2	－	○		－				－	○	－
n	○	○		○				○	－	－
ⓓ	○	○		○				－	－	－

功 能

(1) 从⑤2 中指定的字符串数据的左端的第 n 个字符开始，对⑤1 中指定的字符串数据进行查找后，将查找结果存储到ⓓ中指定的软元件中。

查找结果中存储 ⓢ2 中指定的字符串数据的起始字符算起的第几个字符。

n=3 时



(2) 没有一致的字符串数据的情况下，在 ⓓ 中存储 0。

(3) n 为负数或 0 的情况下，不进行处理。

出 错

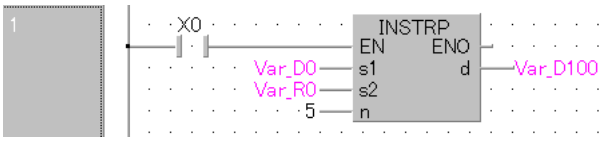
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- n 的值超过了 ⓢ2 的字符数时。 (出错代码： 4100)
- ⓢ1， ⓢ2 中指定的软元件以后，相应软元件范围内没有 00H(NULL) 时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，将 Var_D0 以后存储的字符串数据，从 Var_R0 以后存储的字符串数据的左起第 5 个字符开始查找，并将查找结果存储到 Var_D100 中的程序。

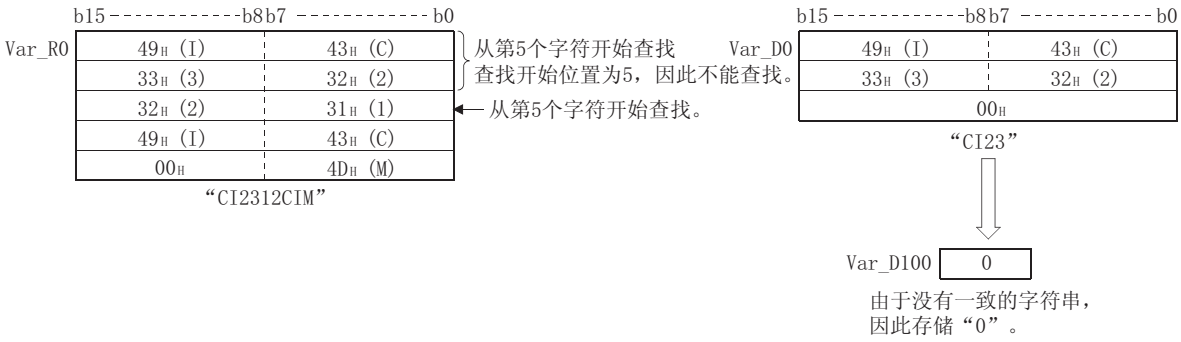
[结构体梯形图]



[ST]

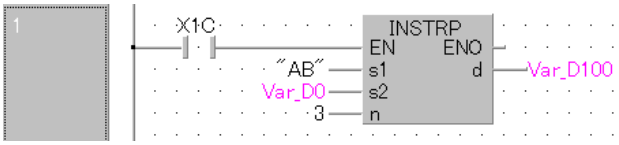
```
INSTRP(X0,Var_D0,Var_R0,5,Var_D100);
```

[动作]



(2) 以下为 X1C 变为 ON 时，将字符串数据 “AB” 从 Var_D0 以后存储的字符串数据的左起第 3 个字符开始查找，并将查找结果存储到 Var_D100 中的程序。

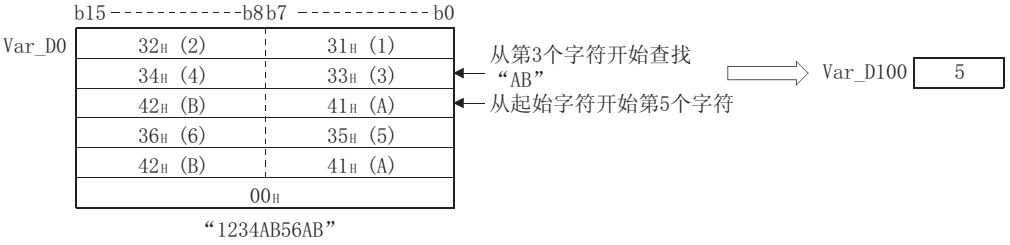
[结构体梯形图]



[ST]

```
INSTRP(X1C,"AB",Var_D0,3,Var_D100);
```

[动作]



7.11.18 字符串插入

STRINS

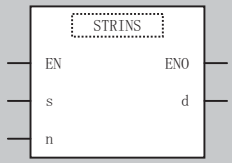
Basic High performance Ver. Universal L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

STRINS(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= STRINS (EN, s, n, d);

[ ] 中放入下述指令。

STRINS　　STRINSP

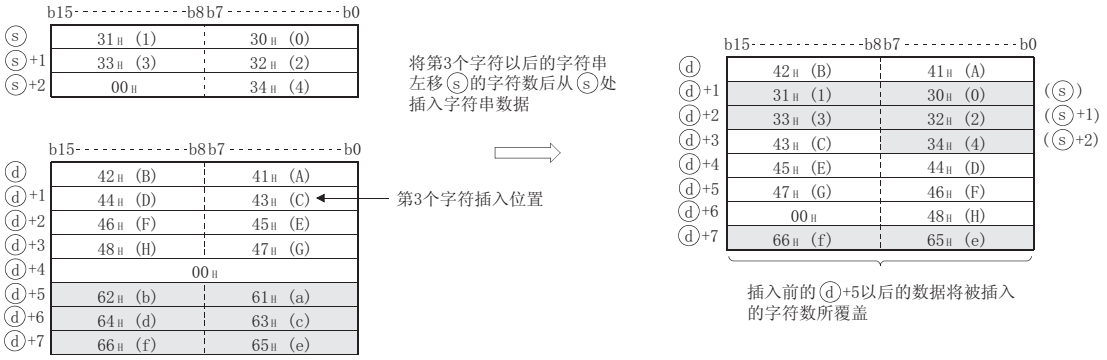
输入自变量，	EN:	执行条件	：位
	s:	插入字符串或存储插入字符串的软元件的起始编号	：字符串
	n:	插入位置（设置范围 1 ≦ n ≦ 16383）	：ANY16
输出自变量，	ENO:	执行结果	：位
	d:	存储插入字符串的软元件的起始编号	：字符串

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K, H	$	
ⓢ1	－	○		－				－	○	－
n	－	○		○				○	－	－
ⓓ	－	○		－				－	－	－

功 能

(1) 将ⓢ中指定的字符串数据，插入到ⓓ中指定的字符串数据的起始算起的第 n 个字符（插入位置）处。

插入位置 n=3 时



(2) 插入后的字符串（ⓢ+ⓓ）为偶数的情况下，字符串的最后的下一个软元件（1 字）处将存储 NULL 码（00H）。

(3) 插入后的字符串（ⓢ+ⓓ）为奇数的情况下，字符串的最后的软元件（高 8 位）处将存储 NULL 码（00H）。

(4) n 中指定了ⓓ的字符数 +1 的情况下，在ⓓ的字符串的最后处与ⓢ的字符串合并。

出 错

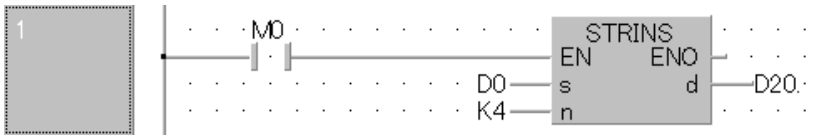
(1) 在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 字符串ⓢ，字符串ⓓ或插入后的字符串（ⓢ+ⓓ）的字符数超过了 16383 个字符时。（出错代码： 4100）
- n 超出了范围时。（1 ≦ n ≦ 16383）（出错代码： 4100）
- n 中指定的值超过了字符串ⓓ的字符数 +1 时。（出错代码： 4100）
- 字符串ⓢ与字符串ⓓ的软元件有部分重叠时。（出错代码： 4101）
- 插入后的字符串（ⓢ+ⓓ）超出了指定软元件的范围时。（出错代码： 4101）
- ⓢ，ⓓ中指定的软元件以后，在指定软元件的范围内 NULL 码（00H）不存在时。（出错代码： 4101）
- 插入后的字符串（ⓢ+ⓓ）与ⓢ的字符串存储软元件重叠时。（出错代码： 4101）

程序示例

(1) 以下为 M0 变为 ON 时，将软元件 D0 以后存储的字符串数据，插入到软元件 D20 以后的字符串数据的起始算起的第 4 个字符处的程序。

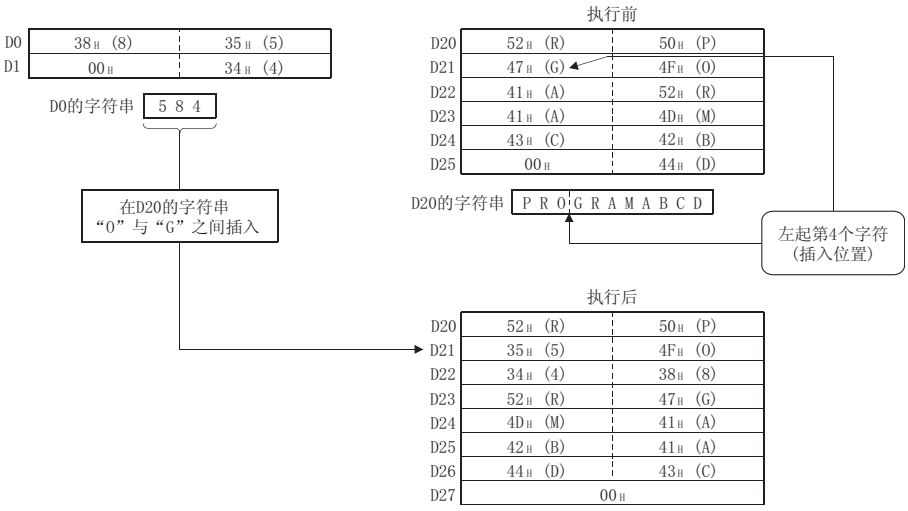
[结构体梯形图]



[ST]

```
STRINS(M0,D0,K4,D20);
```

[动作]



7.11.19 字符串删除

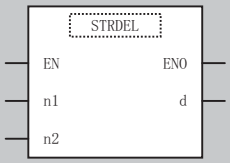
STRDEL

Basic（×） High performance（×） Ver. Universal L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

STRDEL(P)	P：执行条件　　：↑

结构体梯形图



ST

```
ENO:= STRDEL (EN, n1, n2, d);
```

[　　] 中放入下述指令。

STRDEL　　STRDELP

输入自变量，EN：执行条件　：位
n1：删除开始位置（设置范围 1 ≦ n1 ≦ 16383）　：ANY16
n2：删除字符数（设置范围 0 ≦ n2 ≦ 16384-n1）　：ANY16
输出自变量，ENO：执行结果　：位
d：存储删除的字符串的软元件的起始编号　：字符串

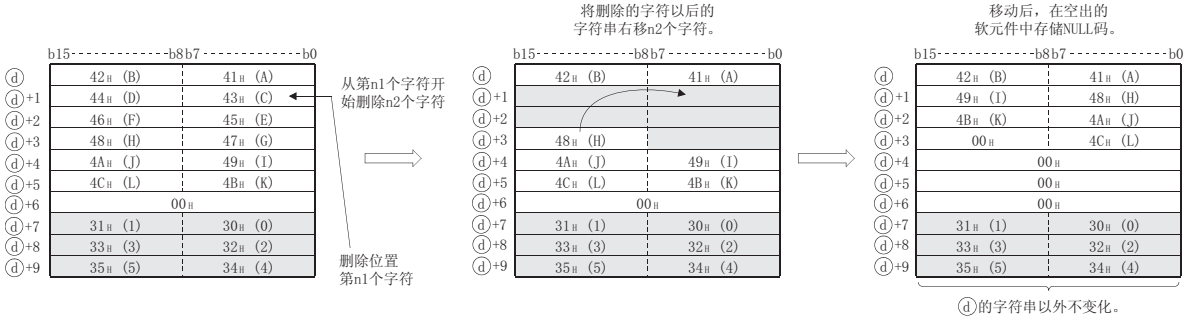
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K, H	
n1	–	○				○		○	–
n2	–	○				○		○	–
ⓓ	–	○				–		–	–

功 能

(1) 从ⓓ中指定的字符串数据的起始算起的第 n1 个字符所指定的位置（删除开始位置）开始，删除 n2 个字符。

删除位置： n=3

删除字符数： n=5 时



(2) 删除后，字符串ⓓ为偶数的情况下，在字符串的最后的下一个软元件（1 字）出存储 NULL 码 (00H)。

(3) 删除后，字符串ⓓ为奇数的情况下，在字符串的最后的软元件（高 8 位）处存储 NULL 码 (00H)。

(4) 将删除后的字符串以后的字符串右移 n2 个字符后，在空出的软元件中存储 NULL 码 (00H)。

出 错

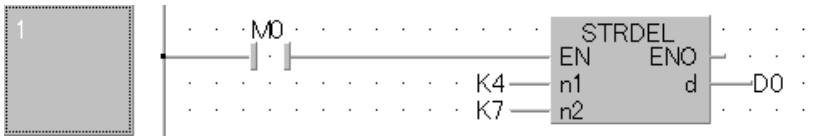
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 字符串ⓓ的字符数超过了 16383 个字符时。 （出错代码： 4100）
- n1 超出了范围时。(1 ≦ n ≦ 16383) （出错代码： 4100）
- n1 中指定的值超过了字符串ⓓ的字符数时。 （出错代码： 4100）
- n2 中指定的值超过了字符串ⓓ的 n1 开始至最终字符为止的字符数时。 （出错代码： 4100）
- n2 中指定的值为负数时。 （出错代码： 4100）

程序示例

以下为 M0 变为 ON 时，从软元件 D0 以后存储的字符串数据的第 4 个字符开始删除 7 个字符的程序。

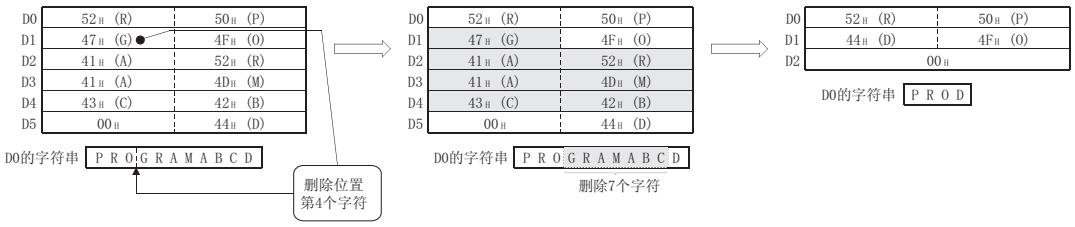
[结构体梯形图]



[ST]

```
STRDEL(M0,K4,K7,D0);
```

[动作]



7.11.20 浮动小数点→ BCD 分解

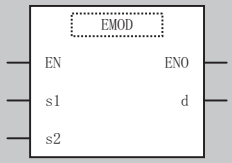
EMOD

Basic　High performance　Universal　L CPU

EMOD(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:=  EMOD  (EN,s1, s2, d);
```

[　　] 中放入下述指令。

EMOD	EMODP

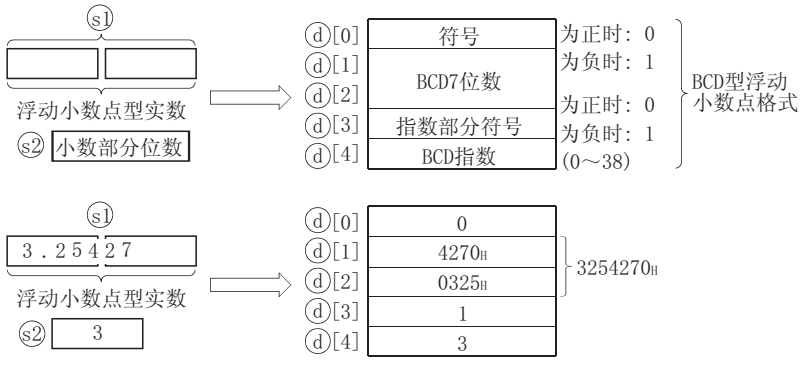
输入自变量，EN：执行条件　　：位
　　s1：浮动小数点型实数数据　　：单精度实数
　　s2：小数部分位数据　　：ANY16
输出自变量，ENO：执行结果　　：位
　　d：BCD 分解后的数据　　：ANY16 的数组（0..4）

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数		其它
	位	字		位	字			K，H	E	
ⓢ1	－	○		－	○		－		○	－
ⓢ2	○	○		○	○		○		－	－
ⓓ	－	○		－	－		－		－	－

功 能

(1) 将ⓢ1 中指定的浮动小数点型实数数据，根据ⓢ2 中指定的小数部分位数，分解为 BCD 型浮动小数点格式后，存储到ⓓ中指定的软元件编号以后。

ⓢ1 中可指定的范围为 0 或 $2^{-126} \sim 2^{128}$。

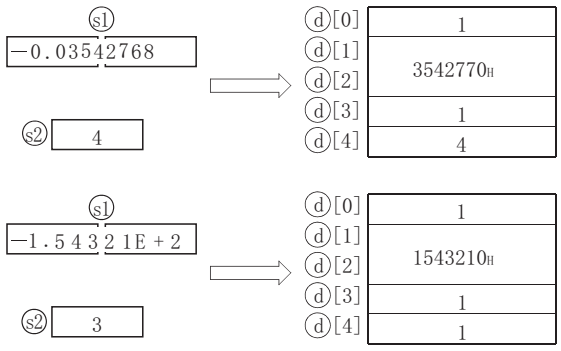


在 ⓢ2 中指定 ⓢ1 的浮动小数点型实数数据的小数部分位数。其情况如下图所示。

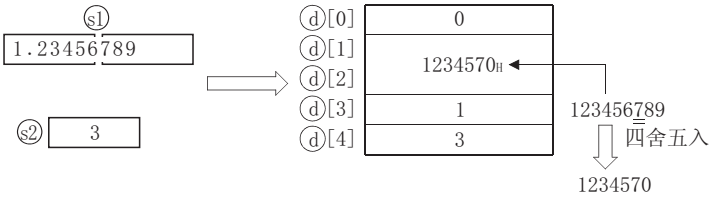
ⓢ2 中可指定的范围为 0 ～ 7。

3.25427

ⓢ2 =3



(2) ⓓ +1, ⓓ +2 中存储的 BCD 的有效位数是将第 7 位进行四舍五入后的 6 位。



出 错

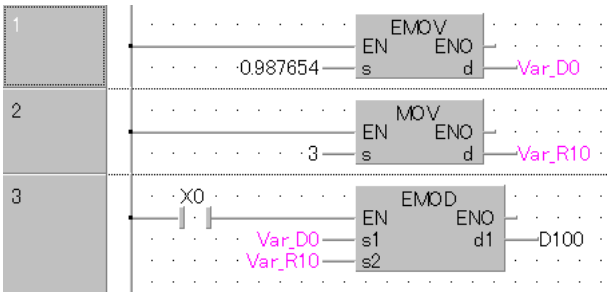
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ2 中指定的小数部分位数为 0 ～ 7 以外时。 (出错代码： 4100)
- ⓓ 中指定的软元件范围超出了相应软元件的范围时。 (出错代码： 4101)
- ⓢ1 中指定的 32 位浮动小数点型实数超出以下范围时。
  0, $2^{-126} \leqq |$指定软元件的内容$| < 2^{128}$ (出错代码： 4100)
- ⓓ 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  (通用型 QCPU、LCPU 时) (出错代码： 4140)

程序示例

以下为 X0 变为 ON 时，将 Var_D0 中存储的浮动小数点型实数数据，根据 Var_R10 中存储的值的小数部分位数分解为 BCD 后，存储到 D100 以后的程序。

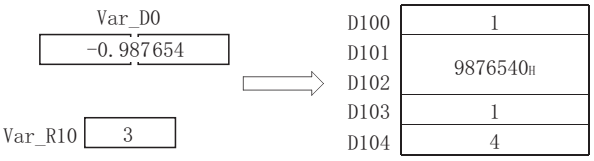
[结构体梯形图]



[ST]

```
Var_D0:=-0.987654;
Var_R10:=3;
EMOD(X0,Var_D0,Var_R10,D100);
```

[动作]



7.11.21 BCD 格式数据→浮动小数点

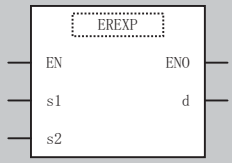
EREXP

Basic | High performance | Universal | L CPU

EREXP(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EREXP (EN, s1, s2, d);

[ ] 中放入下述指令。

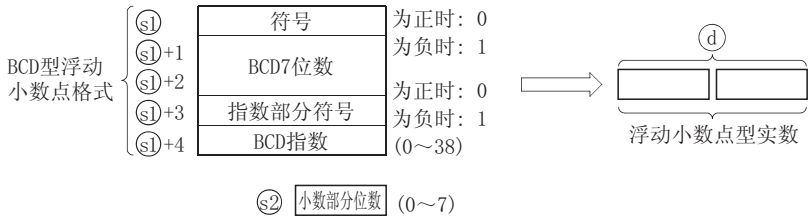
EREXP　　EREXPP

输入自变量，EN:　执行条件　：位
　　s1:　存储 BCD 型浮动小数点格式数据的软元件的起始编号　：ANY16
　　s2:　小数部分位数据　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　浮动小数点型实数数据　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ1	-	○		-	-		-		-
ⓢ2	○	○		○	○		○		-
ⓓ	-	○		-	○		-		-

功 能

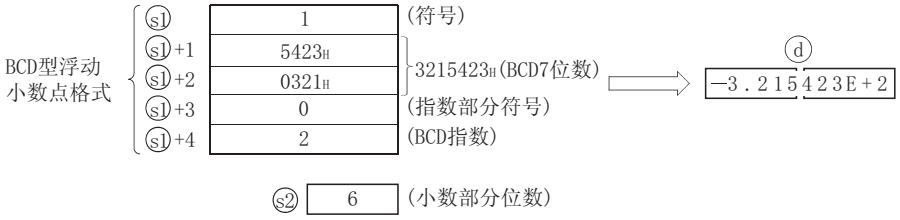
(1) 将ⓢ1中指定的 BCD 型浮动小数点格式数据，根据ⓢ2中指定的小数部分位数转换为浮动小数点型实数数据后，存储到ⓓ中指定的软元件编号以后。



(2) ⓢ1的符号及ⓢ1+3 的指数部分符号中，为正时设置 0，为负时设置 1。

(3) ⓢ1+4 的 BCD 指数中可设置的范围为 0 ～ 38。

(4) ⓢ2 的小数部分位数中可设置的范围为 0 ～ 7。



出 错

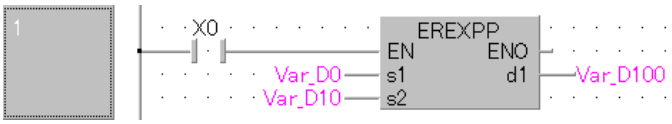
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1 中指定的格式指定为除 0、1 以外时。 (出错代码： 4100)
- ⓢ1 +1， ⓢ1 +2 中指定的 BCD 数据超过了 8 位时。 (出错代码： 4100)
- ⓢ1 +3 中指定的格式指定为除 0、1 以外时。 (出错代码： 4100)
- ⓢ1 +4 中指定的指数数据超出了 0 ～ 38 的范围时。 (出错代码： 4100)
- ⓢ2 中指定的小数部分位数超出了 0 ～ 7 的范围时。 (出错代码： 4100)
- ⓢ1 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

程序示例

以下为 X0 变为 ON 时，将 Var_D0 以后存储的 BCD 型浮动小数点格式数据，根据 Var_D10 中存储的小数部分位数转换为浮动小数点型实数数据后，存储到 Var_D100 中的程序。

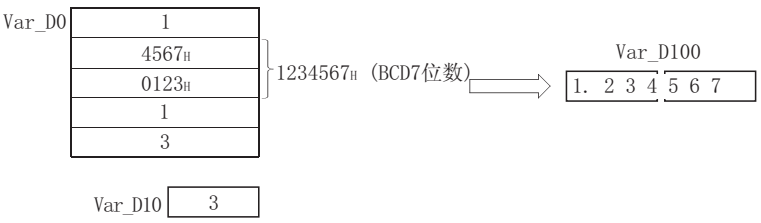
[结构体梯形图]



[ST]

```
EREXPP(X0,Var_D0,Var_D10,Var_D100);
```

[动作]



7.12 特殊函数指令

7.12.1 浮动小数点 SIN 运算（单精度））

SIN

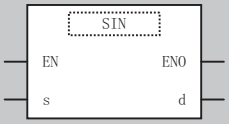
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

SIN(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SIN (EN, s, d);

[ ] 中放入下述指令。

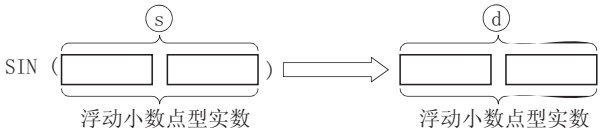
SIN　　SINP

输入自变量，EN:　执行条件　：位
　　s:　进行 SIN（正弦）运算的角度数据　：单精度实数 /ANY32
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	-	○		-	○		-	○	-
ⓓ	-	○		-	○		-	-	-

功 能

(1) 进行ⓢ中指定角度的 SIN（正弦）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) 将ⓢ中指定的角度以弧度单位（角度×π÷ 180）进行设置。
关于角度↔弧度转换，请参阅 RAD(P) 指令、DEG(P) 指令。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

指定软元件的内容为 -0 时。*1
（基本型 QCPU、高性能型 QCPU 时）
（出错代码： 4100）

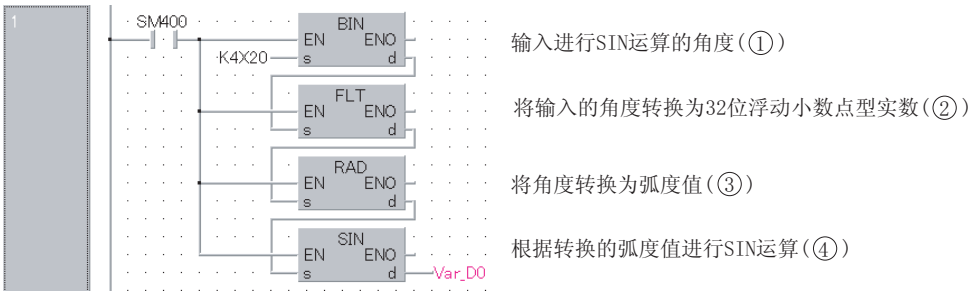
*1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 运算结果超出了下述范围时。（发生了上溢时。）
（通用型 QCPU、LCPU 时）
$2^{128} \leqq |$ 运算结果 $|$ （出错代码： 4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
（通用型 QCPU、LCPU 时） （出错代码： 4140）

程序示例

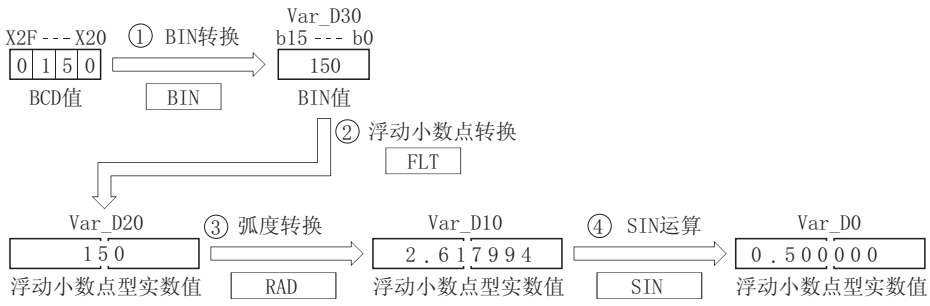
以下为对 X20 ～ X2F 中以 BCD4 位设置的角度进行 SIN 运算后，以浮动小数点型实数存储到 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D30);
     FLT(TRUE,Var_D30,Var_D20);
     RAD(TRUE,Var_D20,Var_D10);
     SIN(TRUE,Var_D10,Var_D0);
END_IF;
```

[在 X20 ～ X2F 中指定了 150 时的动作]



7.12.2 浮动小数点 SIN 运算（双精度）

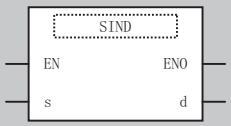
SIND

Basic High performance Universal L CPU

SIND(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SIND (EN, s, d);

中放入下述指令。

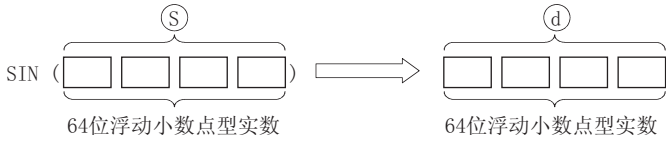
SIND　　SINDP

输入自变量，EN:　执行条件　：位
　　s:　进行 SIN（正弦）运算的角度数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

(1) 进行ⓢ中指定的角度的 SIN（正弦）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) 将ⓢ中指定的角度以弧度单位（角度×π÷180）进行设置。

可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。
关于角度↔弧度转换，请参阅 RADD 指令、DEGD 指令。

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

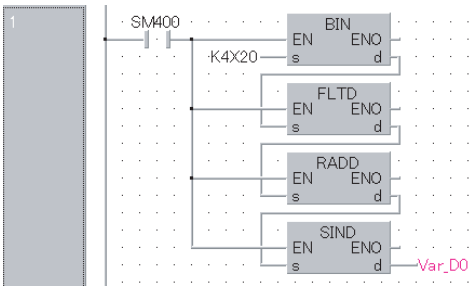
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$运算结果$|$（出错代码： 4141）

程序示例

以下为对 X20 ～ X2F 中以 BCD 4 位设置的角度进行 SIN 运算后，以 64 位浮动小数点型实数存储到 Var_D0 中的程序。

［结构体梯形图］



输入进行SIN运算的角度(①)

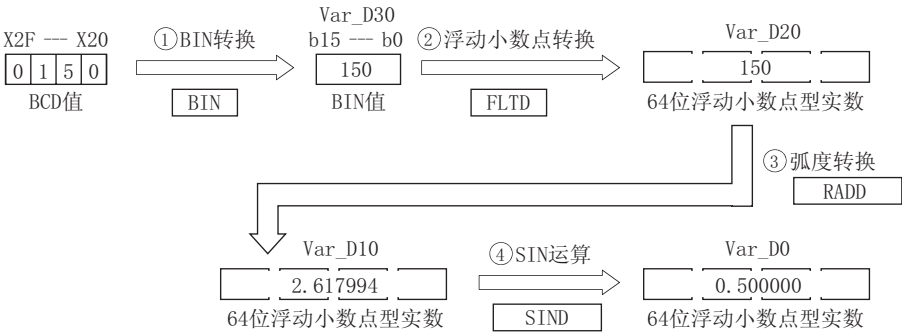
将输入的角度转换为64位浮动小数点型实数(②)

将角度转换为弧度值(③)

根据转换的弧度值进行IN运算(④)

```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D30);
     FLTD(TRUE,Var_D30,Var_D20);
     RADD(TRUE,Var_D20,Var_D10);
     SIND(TRUE,Var_D10,Var_D0);
END_IF;
```

［在 X20 ～ X2F 中指定了 150 时的动作］



7.12.3 浮动小数点 COS 运算（单精度）

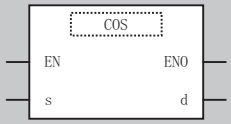
COS

Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

COS(P)	P：执行条件　　：↑

结构体梯形图



ST

```
ENO:=  COS  (EN, s, d);
```

[ ] 中放入下述指令。

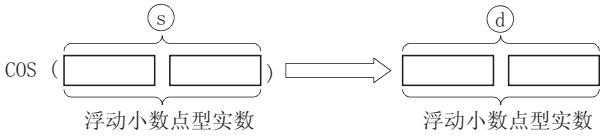
COS　　COSP

输入自变量，	EN:	执行条件	：位
	s:	进行 COS（余弦）运算的角度数据	：单精度实数 /ANY32
输出自变量，	ENO:	执行结果	：位
	d:	运算结果	：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	–	○		–	○		–	○	–
ⓓ	–	○		–	○		–	–	–

功 能

(1) 进行ⓢ中指定的角度的 COS（余弦）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) 将ⓢ中指定的角度以弧度单位（角度×π ÷ 180）进行设置。
关于角度↔弧度转换，请参阅 RAD(P) 指令、DEG(P) 指令。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时
  （出错代码： 4100）

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

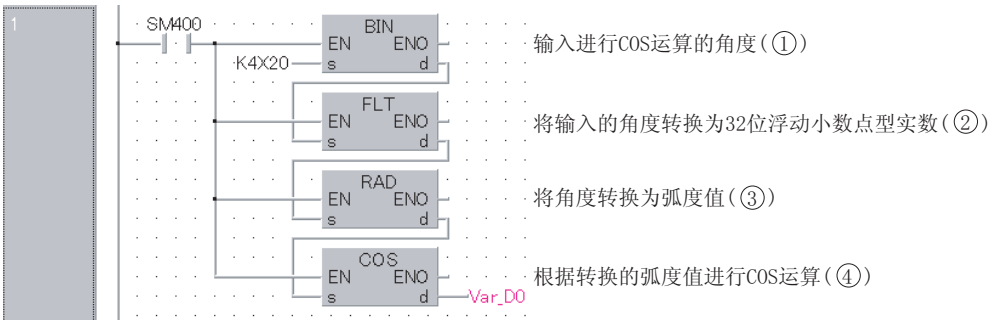
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  $2^{128} \leqq |$运算结果$|$ （出错代码： 4141）

- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码： 4140）

程序示例

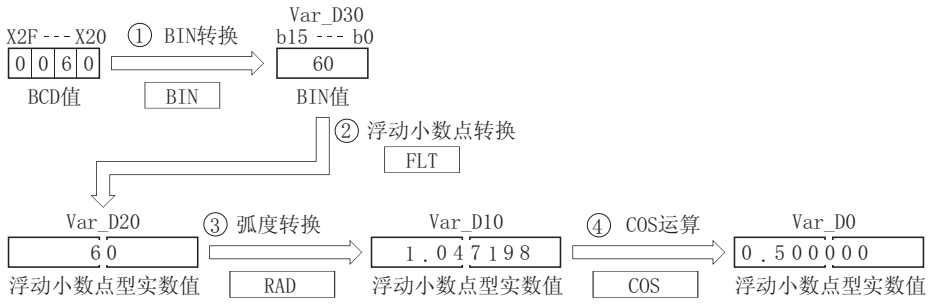
以下为对 X20 ～ X2F 中以 BCD4 位设置的角度进行 COS 运算后，以浮动小数点型实数存储到 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D30);
     FLT(TRUE,Var_D30,Var_D20);
     RAD(TRUE,Var_D20,Var_D10);
     COS(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 60 时的动作]



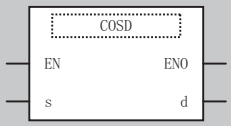
7.12.4 浮动小数点 COS 运算（双精度）

COSD

COSD(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= COSD (EN, s, d);
```

□ 中放入下述指令。

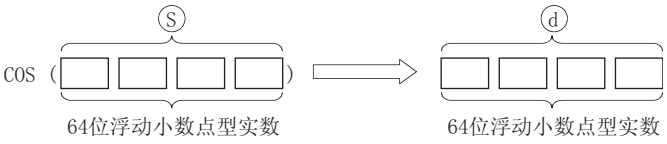
COSD　　COSDP

输入自变量，EN：　执行条件　　：位
　　s：　进行 COS（余弦）运算的角度数据　　：双精度实数
输出自变量，ENO：　执行结果　　：位
　　d：　运算结果　　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

(1) 进行ⓢ中指定的角度的 COS（余弦）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) 将ⓢ中指定的角度以弧度单位（角度×π÷180）进行设置。

可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。
关于角度↔弧度转换，请参阅 RADD 指令、DEGD 指令。

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

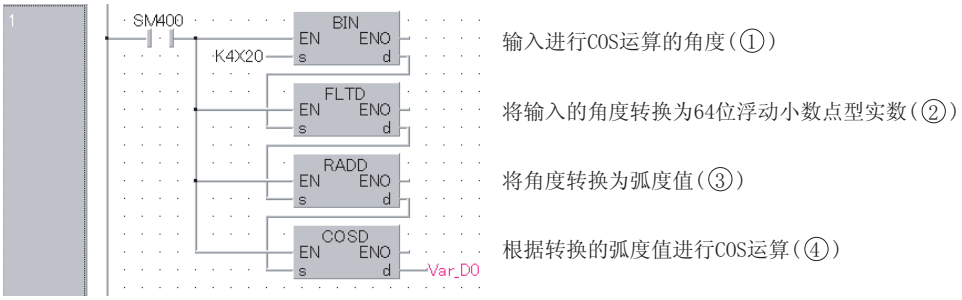
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$（出错代码： 4141）

程序示例

以下为对 X20 ～ X2F 中以 BCD4 位设置的角度进行 COS 运算后，以 64 位浮动小数点型实数存储到 Var_D0 中的程序。

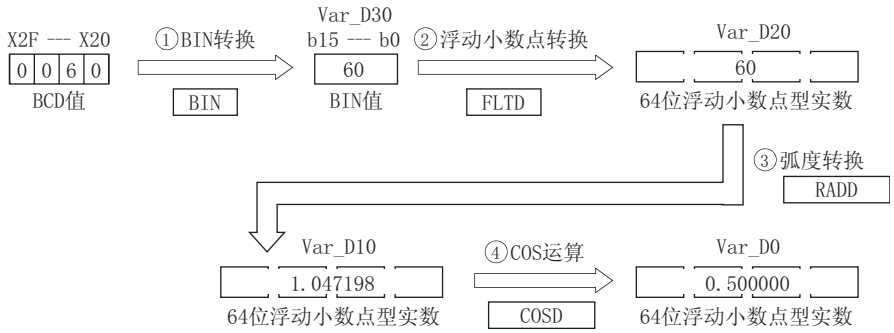
[结构体梯形图]



[ST]

```
IF SM400 THEN
    BIN(TRUE,K4X20,Var_D30);
    FLTD(TRUE,Var_D30,Var_D20);
    RADD(TRUE,Var_D20,Var_D10);
    COSD(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 60 时的动作]



7.12.5 浮动小数点 TAN 运算（单精度）

TAN

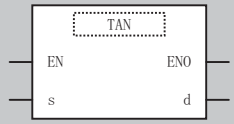
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

TAN(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= TAN (EN, s, d);

[ ] 中放入下述指令。

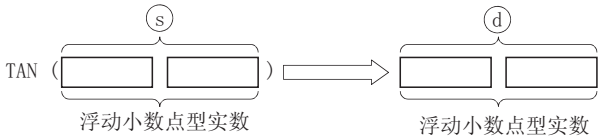
TAN　　TANP

输入自变量，EN：　执行条件　　：位
　　s：　进行 TAN（正切）运算的角度数据　　：单精度实数 /ANY32
输出自变量，ENO：　执行结果　　：位
　　d：　运算结果　　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	-	○		-	○		-	○	-
ⓓ	-	○		-	○		-	-	-

功 能

(1) 进行ⓢ中指定的角度的 TAN（正切）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的角度以弧度单位（角度×π÷180）进行设置。
关于角度↔弧度转换，请参阅 RAD(P) 指令、DEG(P) 指令。

(3) ⓢ中指定的角度为π /2 弧度、(3/2) π弧度的情况下，弧度值中将产生运算误差，但不会变为出错状态，因此应加以注意。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 运算结果超出以下范围时。
  （基本型 QCPU、高性能型 QCPU 时）
  $0, 2^{-126} \leqq |$运算结果$| < 2^{128}$ （出错代码：4100）
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时）
  （出错代码：4100）

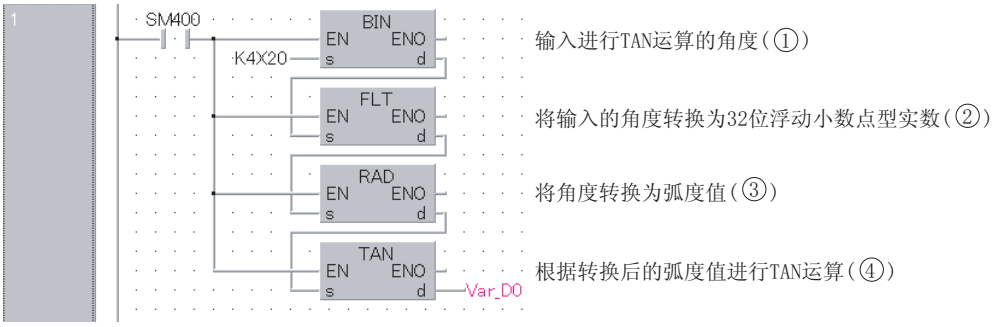
*1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  $2^{128} \leqq |$运算结果$|$ （出错代码：4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码：4140）

程序示例

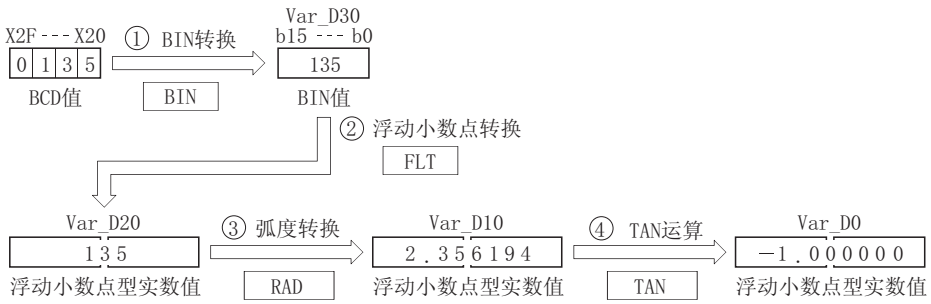
以下为对 X20 ～ X2F 中以 BCD4 位设置的角度进行 TAN 运算后，以浮动小数点型实数存储到 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D30);
     FLT(TRUE,Var_D30,Var_D20);
     RAD(TRUE,Var_D20,Var_D10);
     TAN(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 135 时的动作]



7.12.6 浮动小数点 TAN 运算（双精度）

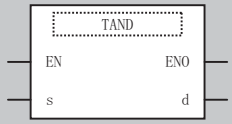
TAND

Basic | High performance | Universal | L CPU

TAND(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= TAND (EN, s, d);

[　　] 中放入下述指令。

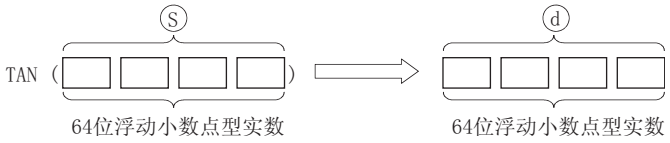
TAND　　TANDP

输入自变量，EN:　执行条件　：位
　　s:　进行 TAN（正切）运算的角度数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 进行ⓢ中指定的角度的 TAN（正切）值运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的角度以弧度单位（角度×π÷180）进行设置。

可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。
关于角度↔弧度转换，请参阅 RADD 指令、DEGD 指令。

(3) ⓢ中指定的角度为 π /2 弧度、(3/2) π 弧度的情况下，弧度值中将产生运算误差，但不会变为出错状态，因此应加以注意。

(4) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

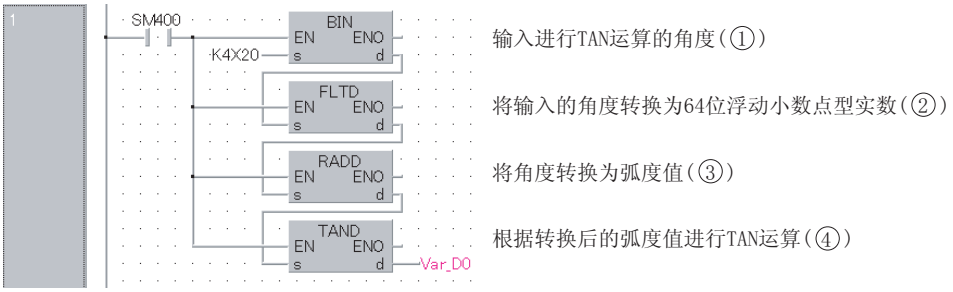
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$（出错代码： 4141）

程序示例

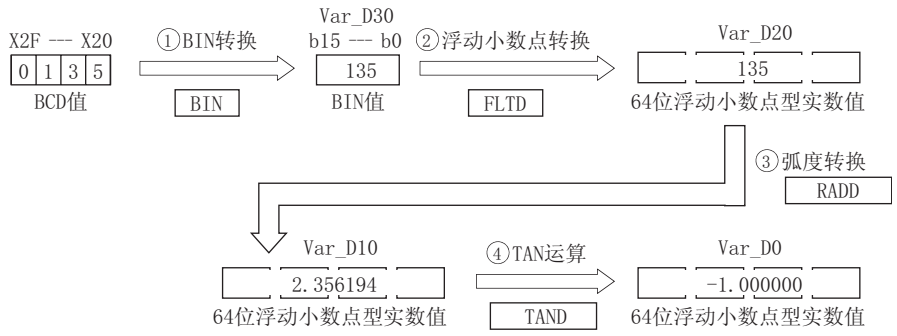
以下为对 X20 ～ X2F 中以 BCD 4 位设置的角度进行 TAN 运算后，以 64 位浮动小数点型实数存储到 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D30);
     FLTD(TRUE,Var_D30,Var_D20);
     RADD(TRUE,Var_D20,Var_D10);
     TAND(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 135 时的动作]



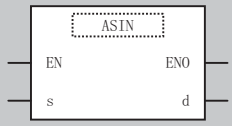
7.12.7 浮动小数点 SIN $^{-1}$ 运算（单精度）

ASIN

Basic | High performance | Universal | L CPU

ASIN(P)	P：执行条件　　：↑

结构体梯形图



ST

ENO:= ASIN (EN, s, d);

□ 中放入下述指令。

ASIN　　ASINP

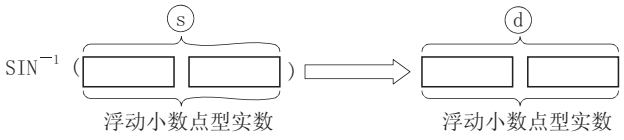
输入自变量，EN:　执行条件　：位
　　s:　进行 SIN^{-1}（反正弦）运算的 SIN 值　：单精度实数 /ANY32
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	–	○		–	○		○ *1	○	–
ⓓ	–	○		–	○		○ *1	–	–

*1 ：仅通用型 QCPU 可以使用。

功 能

(1) 将ⓢ中指定的 SIN 值进行角度运算后，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓢ中指定的 SIN 值的可设置范围为 -1.0 ～ 1.0。

(3) ⓓ中存储的角度（运算结果）以弧度单位进行存储。
关于弧度↔角度转换，请参阅 DEG(P) 指令、RAD(P) 指令。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

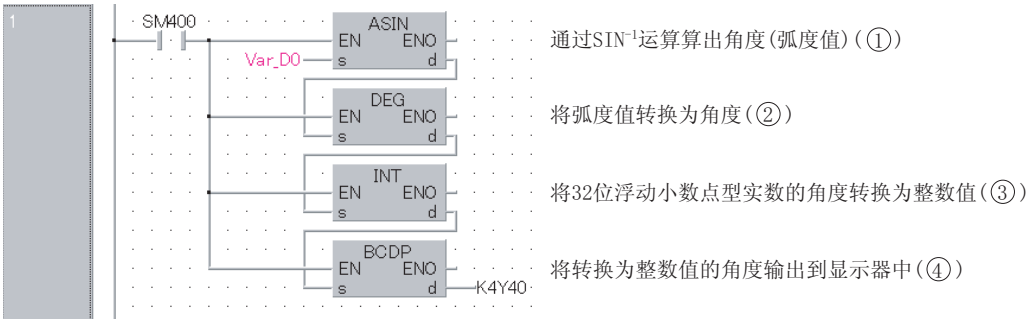
- ⓢ中指定的值超出 -1.0 ～ 1.0 的范围时。（出错代码： 4100）
- 指定软元件的内容超出以下范围时。（通用型 QCPU、LCPU 时）
  0, 2^{-126} ≦ | 指定软元件的内容 | ＜ 2^{128}（出错代码： 4140）
- 指定软元件的内容为 -0 时。*1
  （高性能型 QCPU 时）（出错代码： 4100）

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 |（出错代码： 4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）（出错代码： 4140）

程序示例

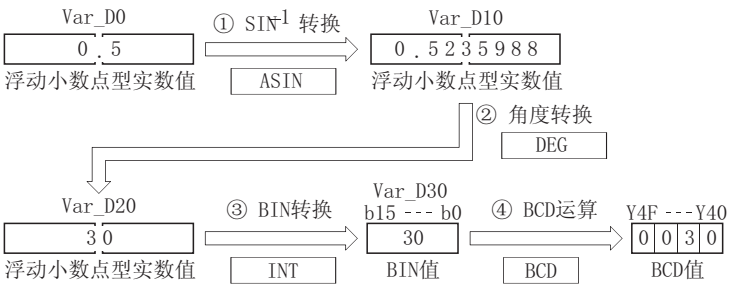
以下为求出 Var_D0 的浮动小数点型实数的 SIN-1 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ASIN(TRUE,Var_D0,Var_D10);
     DEG(TRUE,Var_D10,Var_D20);
     INT(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 0.5 时的动作]



7.12.8 浮动小数点 SIN $^{-1}$ 运算（双精度）

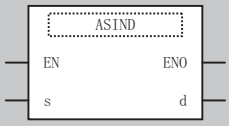
ASIND

Basic High performance Universal L CPU

ASIND(P)

P: 执行条件　　: ⌐

结构体梯形图



ST

```
ENO:= ASIND (EN, s, d);
```

[ ] 中放入下述指令。

ASIND　　ASINDP

输入自变量，EN:　执行条件　：位

s:　进行 SIN^{-1}（反正弦）运算的 SIN 值　：双精度实数

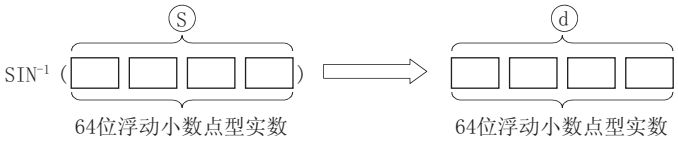
输出自变量，ENO:　执行结果　：位

d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 通过ⓢ中指定的 SIN 值进行角度运算，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓢ中指定的 SIN 值的课可设置范围为 -1.0 ～ 1.0。

(3) ⓓ中存储的角度（运算结果）是以弧度单位存储。
关于弧度↔角度转换，请参阅 DEGD 指令、RADD 指令。

(4) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

7 应用指令

出 错

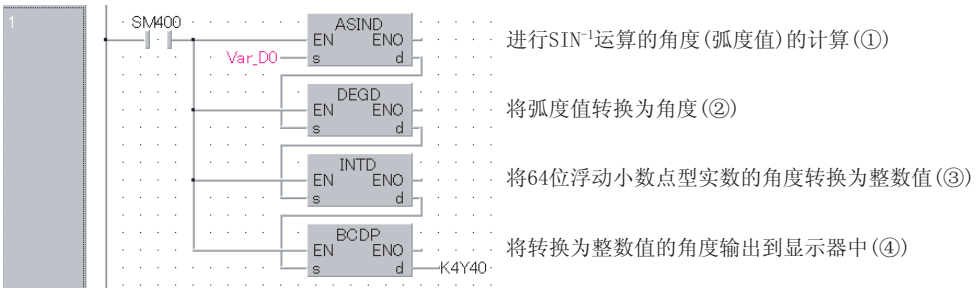
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 （出错代码： 4140）
  0, $2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$ （出错代码： 4141）
- ⓢ 中指定的值超出了 -1.0 ～ 1.0 的范围时。 （出错代码： 4140）

程序示例

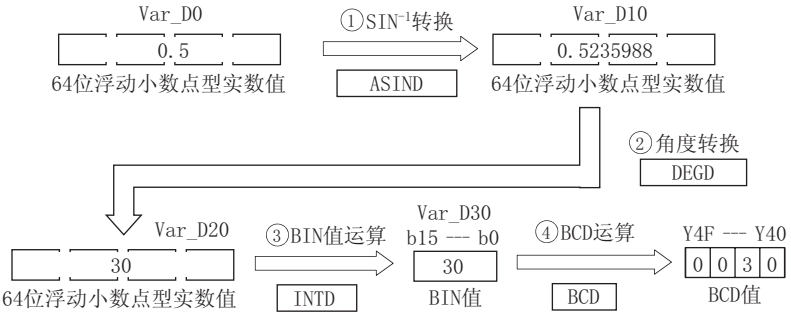
以下为求出 Var_D0 的 64 位浮动小数点型实数的 SIN-1 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ASIND(TRUE,Var_D0,Var_D10);
     DEGD(TRUE,Var_D10,Var_D20);
     INTD(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 0.5 时的动作]



7.12.9 浮动小数点 COS $^{-1}$ 运算（单精度）

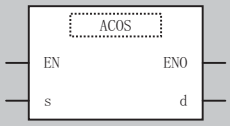
ACOS

Basic　High performance　Universal　L CPU

ACOS(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ACOS (EN, s, d);

[ ] 中放入下述指令。

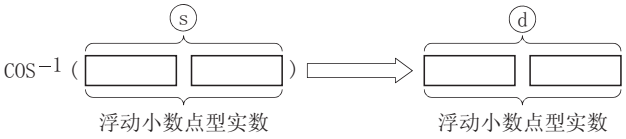
ACOS　　ACOSP

输入自变量，EN：　执行条件　　：位
　　s：　进行 COS^{-1}（反余弦）运算的 COS 值　　：单精度实数 /ANY32
输出自变量，ENO：　执行结果　　：位
　　d：　运算结果　　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	-	○		-	○		-	○	-
ⓓ	-	○		-	○		-	-	-

功 能

(1) 通过ⓢ中指定的 COS 值进行角度运算后，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓢ中指定的 COS 值的可设置范围为 -1.0 ～ 1.0。

(3) ⓓ中存储的角度（运算结果）是以弧度单位存储。
关于弧度↔角度转换，请参阅 DEG(P) 指令、RAD(P) 指令。

出 错

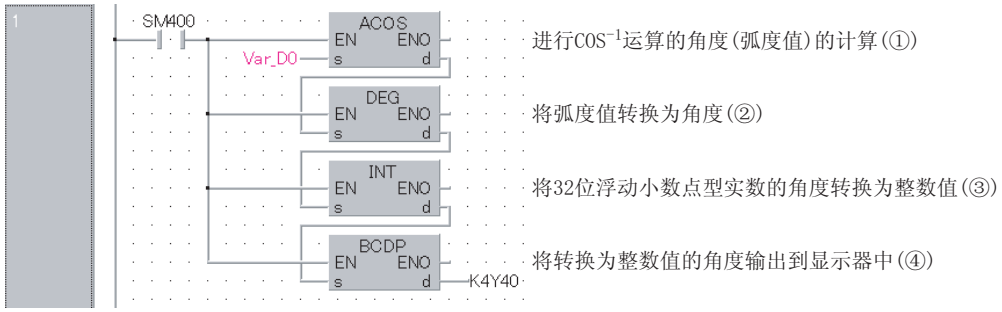
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- Ⓢ 中指定的值超出了 -1.0 ～ 1.0 的范围时。 （出错代码： 4100）
- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  0, 2^{-126} ≦ | 指定软元件的内容 | < 2^{128} （出错代码： 4140）
- 指定软元件的内容为 -0 时。*1 （出错代码： 4100）
  （高性能型 QCPU 时）
  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 | （出错代码： 4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码： 4140）

程序示例

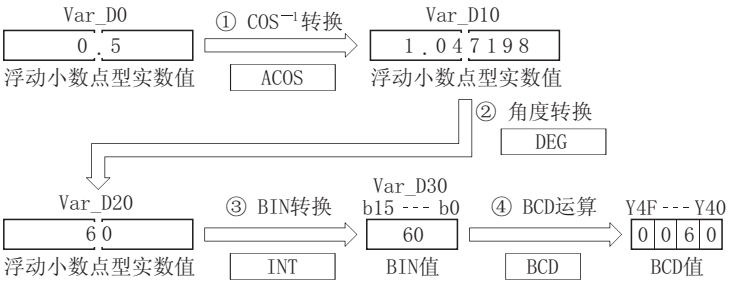
以下为求出 Var_D0 的浮动小数点型实数的 COS^{-1} 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ACOS(TRUE,Var_D0,Var_D10);
     DEG(TRUE,Var_D10,Var_D20);
     INT(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 0.5 时的动作]



7.12.10 浮动小数点 COS $^{-1}$ 运算（双精度）

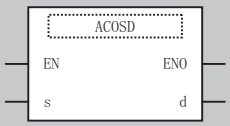
ACOSD

Basic ✗ High performance ✗ Universal L CPU

ACOSD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ACOSD (EN, s, d);

□ 中放入下述指令。

ACOSD　　ACOSDP

输入自变量，EN:　执行条件　：位

s:　进行 COS^{-1}（反余弦）运算的 COS 值　：双精度实数

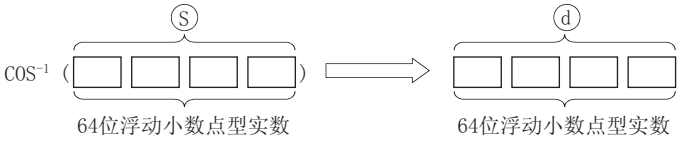
输出自变量，ENO:　执行结果　：位

d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

(1) 通过ⓢ中指定的 COS 值进行角度运算后，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓢ中指定的 COS 值的可设置范围为 -1.0 ～ 1.0。

(3) ⓓ中存储的角度（运算结果）是弧度单位存储。
关于弧度↔角度转换，请参阅 DEGD 指令、RADD 指令。

(4) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

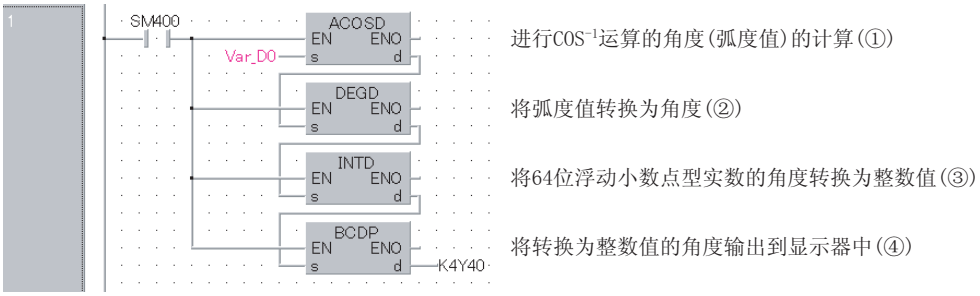
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 (出错代码： 4140)
  0, $2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。 (出错代码： 4140)
- 运算结果超出了下述范围时。(发生了上溢时。)
  $2^{1024} \leqq |$ 运算结果 $|$ (出错代码： 4141)
- ⓢ 中指定的值超出了 -1.0 ～ 1.0 的范围时。 (出错代码： 4100)

程序示例

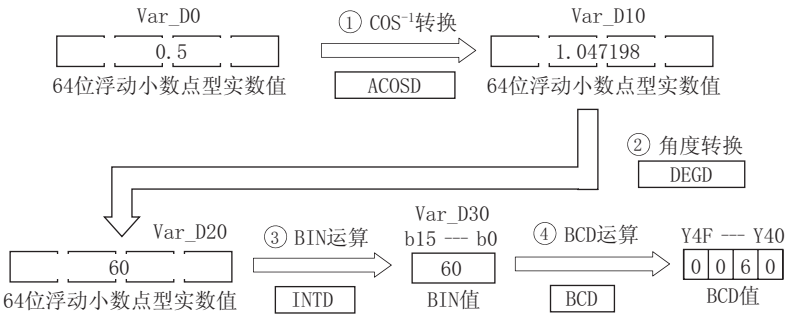
以下为求出 Var_D0 的 64 位浮动小数点型实数的 COS^{-1} 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ACOSD(TRUE,Var_D0,Var_D10);
     DEGD(TRUE,Var_D10,Var_D20);
     INTD(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 0.5 时的动作]



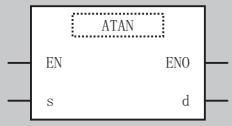
7.12.11 浮动小数点 TAN $^{-1}$ 运算（单精度）

ATAN

Basic | High performance | Universal | L CPU

ATAN(P)	P：执行条件　　：↑

结构体梯形图



ST

ENO:= ATAN (EN, s, d);

□ 中放入下述指令。

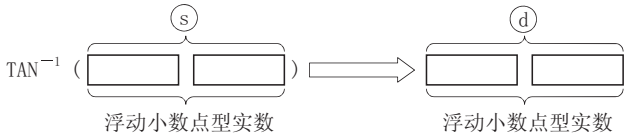
ATAN　　ATANP

输入自变量，EN:　执行条件　：位
　　s:　进行 TAN^{-1}（反正切）运算的 TAN 值　：单精度实数 /ANY32
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－	○		－	○	－
ⓓ	－	○		－	○		－	－	－

功 能

(1) 通过ⓢ中指定的 TAN 值进行角度运算后，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓓ中存储的角度（运算结果）是以弧度单位存储。
关于弧度↔角度转换，请参阅 DEG(P) 指令、RAD(P) 指令。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  0，2^{-126} ≦ | 指定软元件的内容 | < 2^{128} （出错代码：4140）
- 指定软元件的内容为 -0 时。*1
  （高性能型 QCPU 时） （出错代码：4100）

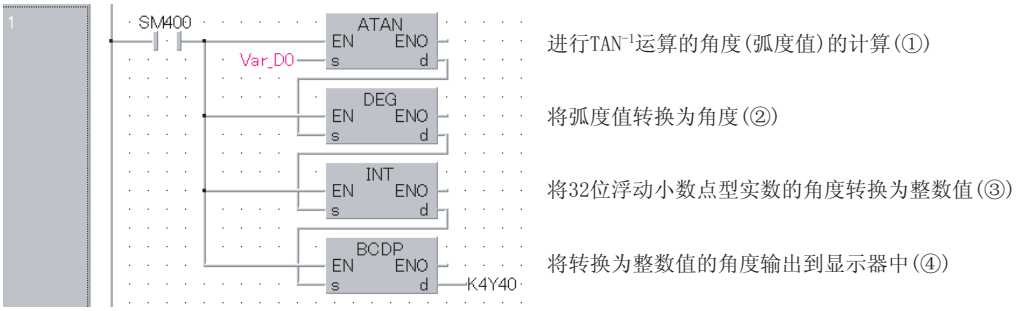
*1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 | （出错代码：4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码：4140）

程序示例

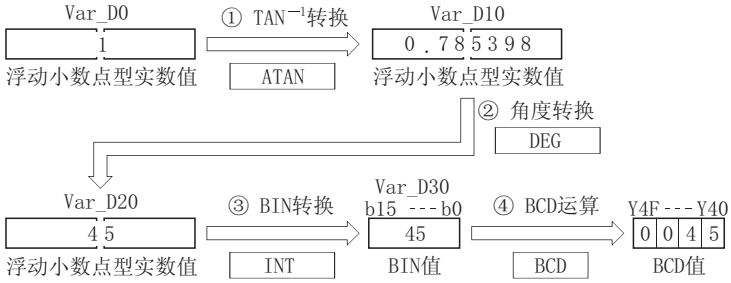
以下为求出 Var_D0 的浮动小数点型实数的 TAN^{-1} 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ATAN(TRUE,Var_D0,Var_D10);
     DEG(TRUE,Var_D10,Var_D20);
     INT(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 1 时的动作]



7.12.12 浮动小数点 TAN $^{-1}$ 运算（双精度）

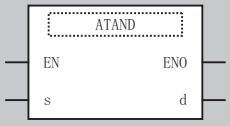
ATAND

Basic　High performance　Universal　L CPU

ATAND(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= ATAND (EN, s, d);

中放入下述指令。

ATAND　　ATANDP

输入自变量，EN:　执行条件　：位

s:　进行 TAN^{-1}（反正切）运算的 TAN 值　：双精度实数

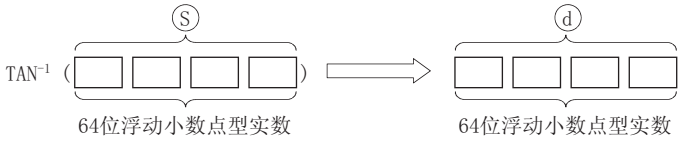
输出自变量，ENO:　执行结果　：位

d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 通过ⓢ中指定的 TAN 进行角度运算后，将运算结果存储到ⓓ中指定的字软元件中。



(2) ⓓ中存储的角度（运算结果）以弧度单位存储。

可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。

关于弧度↔角度转换，请参阅 DEGD 指令、RADD 指令。

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

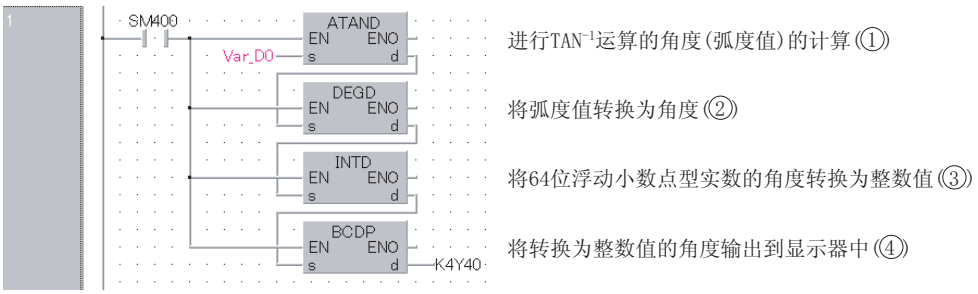
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 （出错代码： 4140）
  0, $2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$ （出错代码： 4141）

程序示例

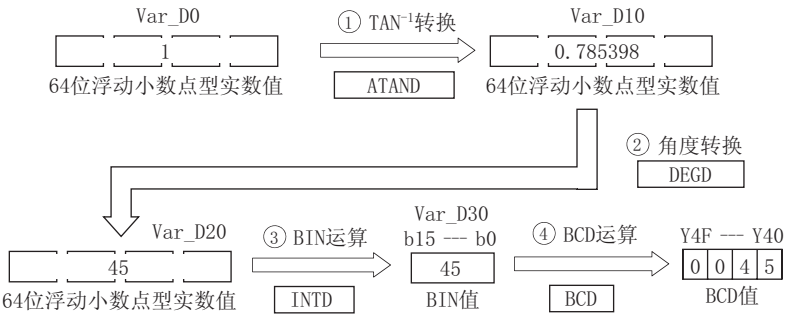
以下为求出 Var_D0 的 64 位浮动小数点型实数的 TAN^{-1} 后，将该角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     ATAND(TRUE,Var_D0,Var_D10);
     DEGD(TRUE,Var_D10,Var_D20);
     INTD(TRUE,Var_D20,Var_D30);
     BCDP(TRUE,Var_D30,K4Y40);
END_IF;
```

[Var_D0 的值为 1 时的动作]



7.12.13 浮动小数点角度→弧度转换（单精度）

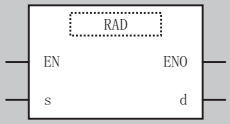
RAD

Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

RAD(P)	P：执行条件　　：↑

结构体梯形图



ST

ENO:= RAD (EN, s, d);

□ 中放入下述指令。

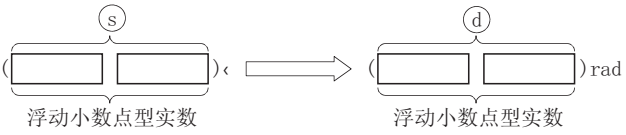
RAD　　RADP

输入自变量，EN：　执行条件　：位
　　s：　转换为弧度单位的角度　：单精度实数 /ANY32
输出自变量，ENO：　执行结果　：位
　　d：　转换为弧度单位后的值　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	–	○		–	○		–	○	–
ⓓ	–	○		–	○		–	–	–

功 能

(1) 将角度的大小单位从ⓢ中指定的度单位转换为弧度单位后，存储到ⓓ中指定的软元件中。



(2) 度单位→弧度单位转换按以下方式进行。

$$\text{弧度单位} = \text{度单位} \times \frac{\pi}{180}$$

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

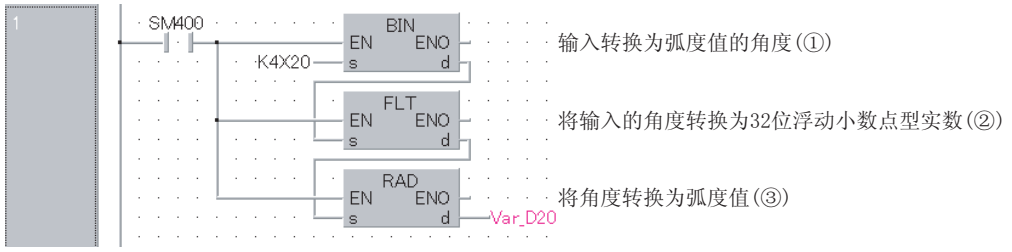
- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  0, 2^{-126} ≦ | 指定软元件的内容 | < 2^{128} （出错代码： 4140)
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时） （出错代码： 4100)

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 | （出错代码： 4141)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码： 4140)

程序示例

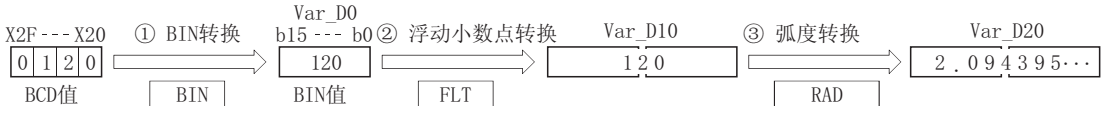
以下为将 X20 ～ X2F 中以 BCD 4 位设置的角度转换为弧度后，以浮动小数点型实数存储到 Var_D20 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D0);
     FLT(TRUE,Var_D0,Var_D10);
     RAD(TRUE,Var_D10,Var_D20);
END_IF;
```

[X20 ～ X2F 中指定了 120 时的动作]



7.12.14 浮动小数点角度→弧度转换（双精度）

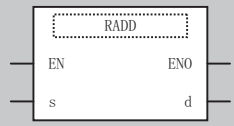
RADD

Basic High performance Universal L CPU

RADD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= RADD (EN, s, d);

□ 中放入下述指令。

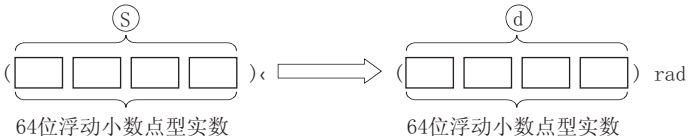
RADD　　RADDP

输入自变量，EN：　执行条件　　：位
　　s：　转换为弧度单位的角度　　：双精度实数
输出自变量，ENO：　执行结果　　：位
　　d：　转换为弧度单位后的值　　：双精度实数

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 将角度的大小单位从ⓢ中指定的度单位转换为弧度单位后，存储到ⓓ中指定的软元件中。
ⓢ中可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。



(2) 度单位→弧度单位转换按以下方式进行。

$$\text{弧度单位} = \text{度单位} \times \frac{\pi}{180}$$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

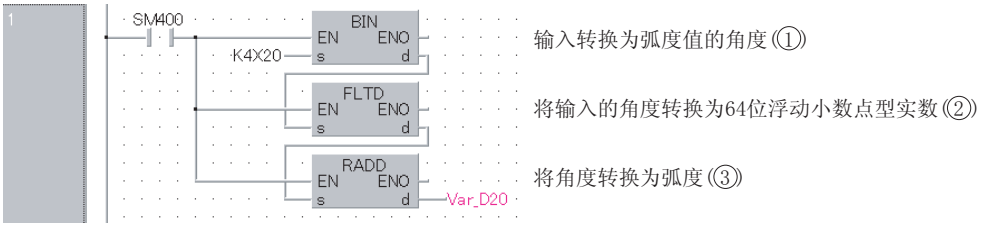
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 （出错代码： 4140）
  0, $2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$ （出错代码： 4141）

程序示例

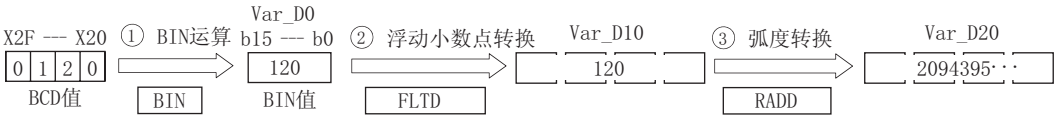
以下为将 X20 ～ X2F 中以 BCD4 位设置的角度转换为弧度后，以 64 位浮动小数点型实数存储到 Var_D20 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D0);
     FLTD(TRUE,Var_D0,Var_D10);
     RADD(TRUE,Var_D10,Var_D20);
END_IF;
```

[X20 ～ X2F 中指定了 120 时的动作]



7.12.15 浮动小数点弧度→角度转换（单精度）

DEG

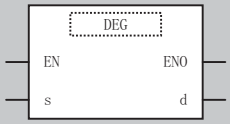
Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

DEG(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= DEG (EN, s, d);

[ ] 中放入下述指令。

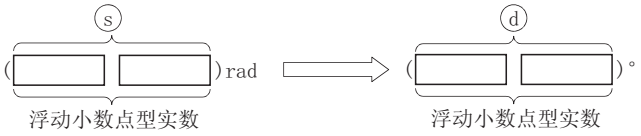
DEG　　DEGP

输入自变量，EN:　执行条件　：位
　　s:　转换为度单位的弧度角度　：单精度实数 /ANY32
输出自变量，ENO:　执行结果　：位
　　d:　转换为度单位后的值　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－	○		－	○	－
ⓓ	－	○		－	○		－	－	－

功 能

(1) 将角度的大小单位从ⓢ中指定的弧度单位转换为度单位后，存储到ⓓ中指定的软元件中。



(2) 度单位→弧度单位转换按以下方式进行。

$$\text{度单位} = \text{弧度单位} \times \frac{180}{\pi}$$

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时）　　（出错代码： 4100）

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。

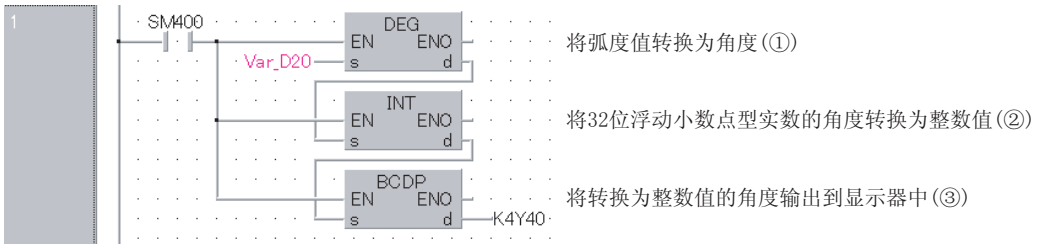
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  $2^{128} \leqq |$ 运算结果 $|$　　（出错代码： 4141）

- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）　　（出错代码： 4140）

程序示例

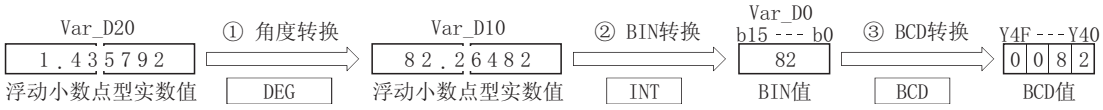
以下为将 Var_D20 中以浮动小数点型实数设置的弧度值转换为角度后，以 BCD 值输出到 Y40 ～ Y4F 中程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     DEG(TRUE,Var_D20,Var_D10);
     INT(TRUE,Var_D10,Var_D0);
     BCDP(TRUE,Var_D0,K4Y40);
END_IF;
```

[Var_D20 的值为 1.435792 时的动作]



7.12.16 浮动小数点弧度→角度转换（双精度）

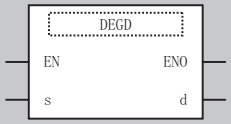
DEGD

Basic High performance Universal L CPU

DEGD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= DEGD (EN, s, d);

[ ] 中放入下述指令。

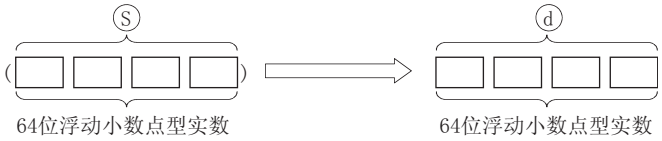
DEGD　　DEGDP

输入自变量，EN:　执行条件　：位
　　s:　转换为度单位的弧度角度　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　转换为度单位后的值　：双精度实数

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 将角度的大小单位从ⓢ中指定的弧度单位转换为度单位后，存储到ⓓ中指定的软元件中。
ⓢ中可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。



(2) 弧度单位→度单位转换按以下方式进行。

$$\text{度单位} = \text{弧度单位} \times \frac{180}{\pi}$$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

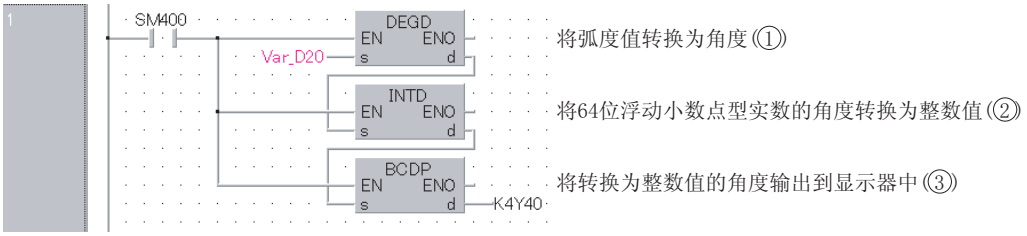
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。 （出错代码： 4140）
  0, $2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。 （出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$ （出错代码： 4141）

程序示例

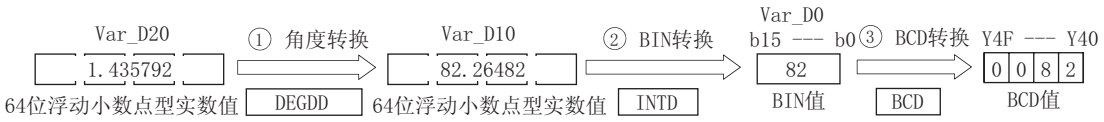
以下为将 Var_D20 中以 64 位浮动小数点型实数设置的弧度值转换为角度后，以 BCD 值输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     DEGD(TRUE,Var_D20,Var_D10);
     INTD(TRUE,Var_D10,Var_D0);
     BCDP(TRUE,Var_D0,K4Y40);
END IF;
```

[Var_D20 的值为 1.435792 时的动作]



7.12.17 浮动小数点幂运算（单精度）

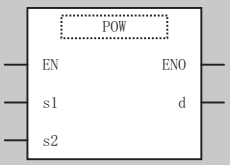
POW

Basic ✗ High performance ✗ Ver. Universal L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

POW(P)	P：执行条件　　：

结构体梯形图



ST

```
ENO:=  POW  (EN, s1, s2, d);
```

[　　] 中放入下述指令。

POW	POWP

输入自变量，EN：执行条件　：位

s1：被幂运算的数据或存储被幂运算的数据的软元件的起始编号　：单精度实数

s2：幂运算数据或存储数据的软元件的起始编号　：单精度实数

输出自变量，ENO：执行结果　：位

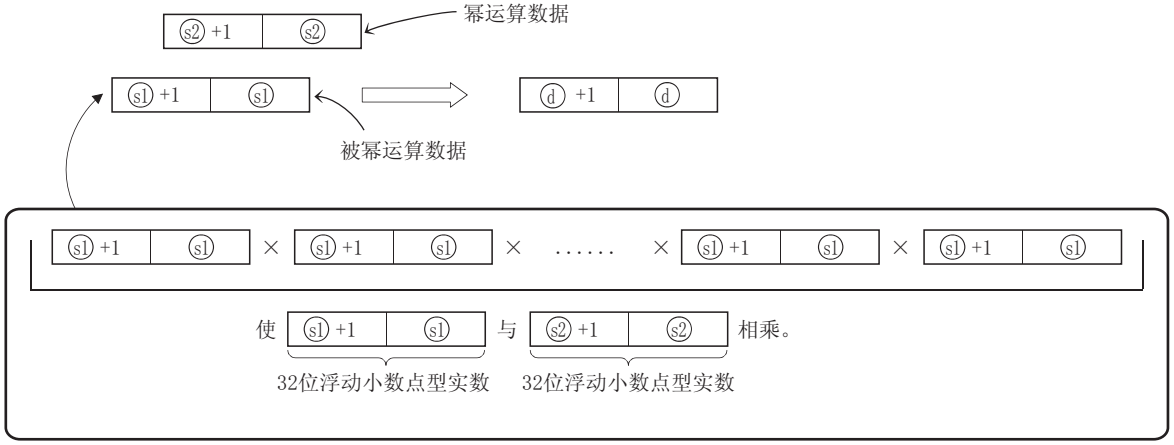
d：存储运算结果的软元件的起始编号　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ1	–	○		–	○			△*1	–
ⓢ2	–	○		–	○			△*1	–
ⓓ	–	○		–	○			–	–

*1：只能使用单精度实数。

功 能

(1) 将 ⓢ1 中指定的 32 位浮动小数点型实数与 ⓢ2 中指定的 32 位浮动小数点型实数进行幂运算后，将运算结果存储到 ⓓ 中指定的软元件中。



(2) ⓢ1, ⓢ2 中可指定的值以及可存储的值如下所示。

0, $2^{-126} \leqq$ | 设置值（存储值）| $< 2^{128}$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1 或 ⓢ2 中指定的值超出以下范围时。（出错代码：4140）

  0, $2^{-126} \leqq$ | 指定值（存储值）| $< 2^{128}$

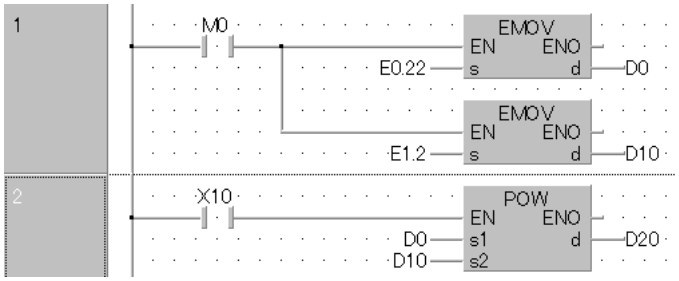
- ⓢ1 或 ⓢ2 的内容为 0 时。（出错代码：4140）
- 运算结果在下述范围内时。（发生了上溢时）

  $2^{128} \leqq$ | 运算结果 |（出错代码：4141）

程序示例

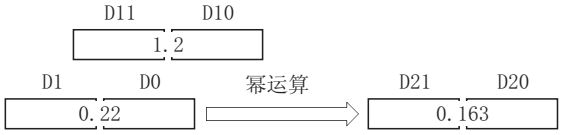
(1) 以下为将 X10 置为 ON 时，将 D0、D1 的 32 位浮动小数点型实数与 D10、D11 的 32 位浮动小数点型实数进行幂运算后，将运算结果存储到 D20、D21 中的程序。

[结构体梯形图]




```
[ST]
IF M0 THEN
     EMOV(TRUE,E0.22,D0);
     EMOV(TRUE,E1.2,D10);
END_IF;
IF X10 THEN
     POW(TRUE,D0,D10,D20);
END_IF;
```

[动作]



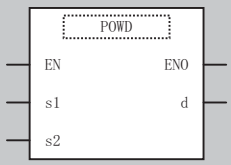
7.12.18 浮动小数点幂运算（双精度）

POWD

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

POWD(P)	P：执行条件　　：↑

结构体梯形图



ST

```
ENO:= POWD (EN, s1, s2, d);
```

[　　] 中放入下述指令。

POWD	POWDP

输入自变量，EN：执行条件　：位

s1：被幂运算的数据或存储被幂运算的数据的软元件的起始编号　：双精度实数

s2：幂运算数据或存储数据的软元件的起始编号　：双精度实数

输出自变量，ENO：执行结果　：位

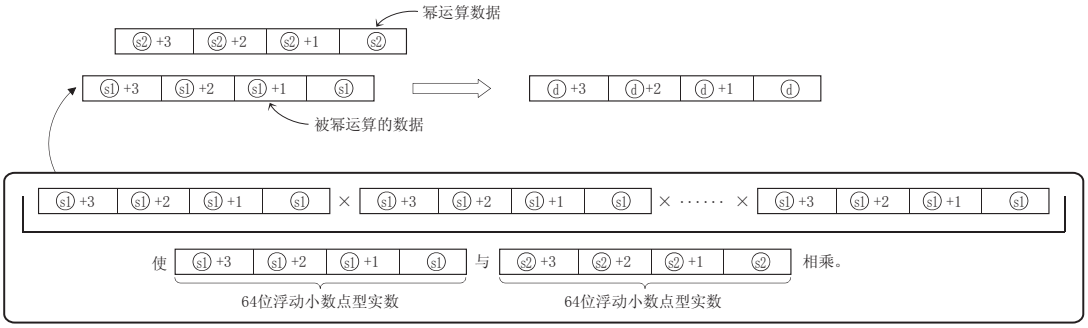
d：存储运算结果的软元件的起始编号　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
⑤1	–	○		–	○			△*1	–
⑤2	–	○		–	○			△*1	–
ⓓ	–	○		–	○			–	–

*1：只能使用双精度实数。

功 能

(1) 将[s1]中指定的 64 位浮动小数点型实数与[s2]中指定的 64 位浮动小数点型实数进行幂运算后，将运算结果存储到[d]中指定的软元件中。



(2) [s1]，[s2]中可指定的值以及可存储的值如下所示。

0，$2^{-1022} \leqq |$ 设置值（存储值）$| < 2^{1024}$

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- [s1]或[s2]中指定的值超出以下范围时。（出错代码： 4140）

  0，$2^{-1022} \leqq |$ 指定值（存储值）$| < 2^{1024}$

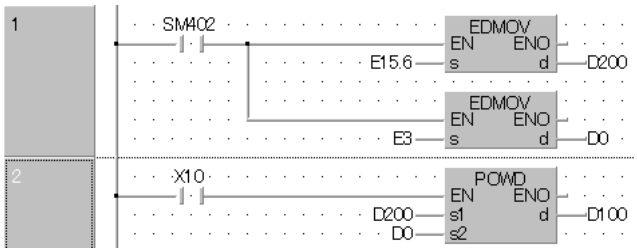
- [s1]或[s2]为 0 时。（出错代码： 4140）
- 运算结果在下述范围内时。（发生了上溢时）

  $2^{1024} \leqq |$ 运算结果 $|$（出错代码： 4141）

程序示例

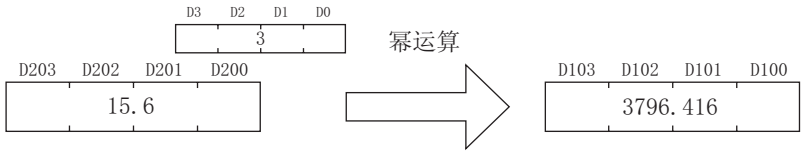
(1) 以下为将 X10 置为 ON 时，将 D200 ～ D203 的 64 位浮动小数点型实数与 D0 ～ D3 的 64 位浮动小数点型实数进行幂运算后，将运算结果存储到 D100 ～ D103 中的程序。

[结构体梯形图]




```
[ST]
IF SM402 THEN
     EDMOV(TRUE,E15.6,D200);
     EDMOV(TRUE,E3,D0);
END_IF;
IF X10 THEN
     POWD(TRUE,D200,D0,D100);
END_IF;
```

[动作]



7.12.19 浮动小数点平方根（单精度）

SQR

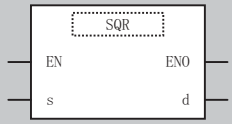
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

SQR(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SQR (EN, s, d);

[ ] 中放入下述指令。

SQR　　SQRP

输入自变量，EN：执行条件　　：位

s：进行平方根运算的数据　　：单精度实数

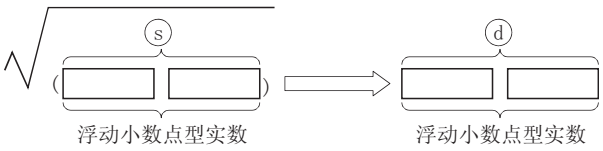
输出自变量，ENO：执行结果　　：位

d：运算结果　　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－	○		－	○	－
ⓓ	－	○		－	○		－	－	－

功 能

(1) 对ⓢ中指定的值进行平方根运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时不能运算。）

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

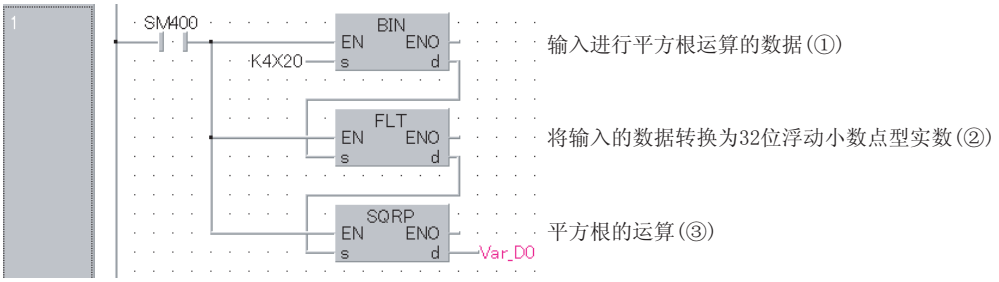
- ⓢ中指定的值为负值时。 （出错代码： 4100）
- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  0, 2^{-126} ≦ | 指定软元件的内容 | < 2^{128} （出错代码： 4140）
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU、高性能型 QCPU 时） （出错代码： 4100）

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 | （出错代码： 4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时） （出错代码： 4140）

程序示例

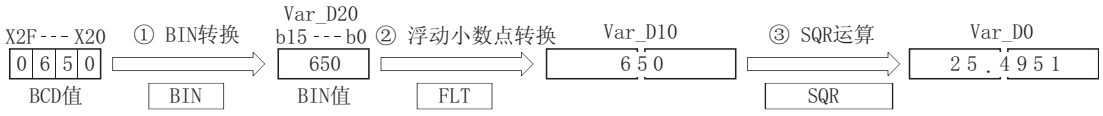
以下为求出 X20 ～ X2F 中以 BCD 4 位设置的值的平方根后，存储到浮动小数点型实数 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     BIN(TRUE,K4X20,Var_D20);
     FLT(TRUE,Var_D20,Var_D10);
     SQRP(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 650 时的动作]



7.12.20 浮动小数点平方根（双精度）

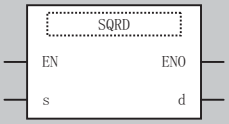
SQRD

Basic High performance Universal L CPU

SQRD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= SQRD (EN, s, d);

中放入下述指令。

SQRD　　SQRDP

输入自变量，EN:　执行条件　：位

s:　进行平方根运算的数据　：双精度实数

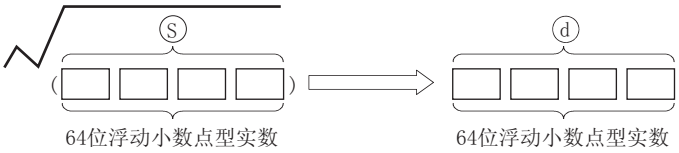
输出自变量，ENO:　执行结果　：位

d:　运算结果　：双精度实数

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 对ⓢ中指定的值进行平方根运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时将不能运算。）

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

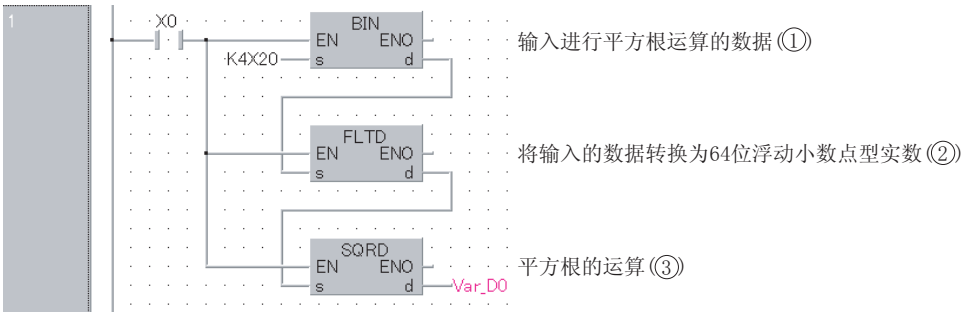
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的值为负值时。 (出错代码： 4100)
- 指定软元件的内容超出以下范围时。 (出错代码： 4140)
  0, $2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。 (出错代码： 4140)
- 运算结果超出了下述范围时。(发生了上溢时。)
  $2^{1024} \leqq |$ 运算结果 $|$ (出错代码： 4141)

程序示例

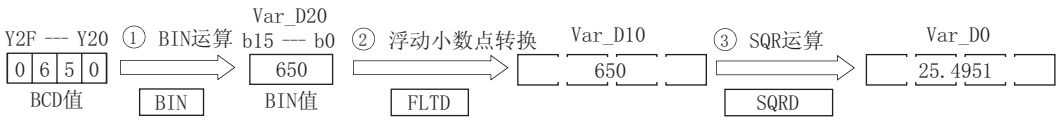
以下为求出 X20 ～ X2F 中以 BCD4 位设置的值的平方根后，存储到 64 位浮动小数点型实数 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
IF X0 THEN
     BIN(TRUE,K4X20,Var_D20);
     FLTD(TRUE,Var_D20,Var_D10);
     SQRD(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X2F 中指定了 650 时的动作]



7.12.21 浮动小数点指数运算（单精度）

EXP

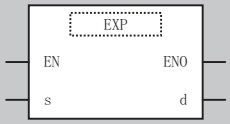
Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

EXP(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= EXP (EN, s, d);

[ ] 中放入下述指令。

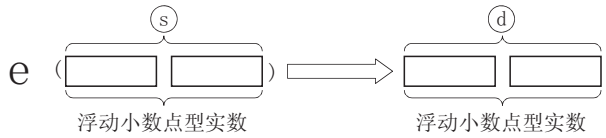
EXP　　EXPP

输入自变量，EN:　执行条件　：位
　　s:　进行指数运算的数据　：单精度实数 /ANY32
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	–	○		–	○		–	○	–
ⓓ	–	○		–	○		–	–	–

功 能

(1) 对ⓢ中指定的值进行指数运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) 在指数运算中，将底 (e) 设置为 2.71828 进行运算。

出 错

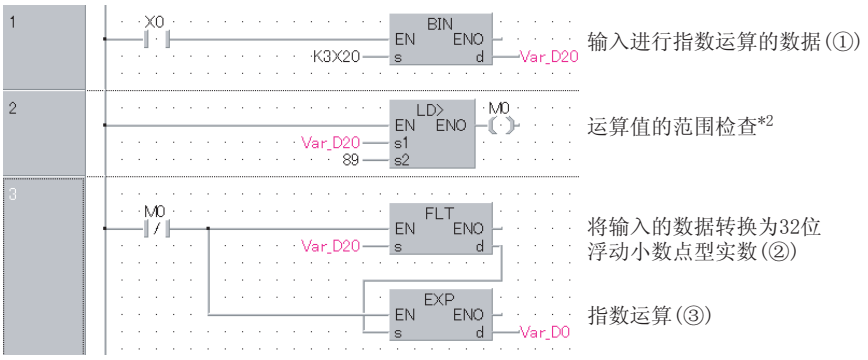
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 运算结果不在下述范围内时。 (出错代码： 4100)
  2^{-126}≦ | 运算结果 | ≦2^{128} (高性能型 QCPU 时)
  2^{-126}≦ | 运算结果 | <2^{128} (基本型 QCPU 时)
- 指定软元件的内容为 -0 时。*1
  (基本型 QCPU，高性能型 QCPU 时) (出错代码： 4100)
  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册 (基础篇)。
- 运算结果超出了下述范围时。(发生了上溢时。)
  (通用型 QCPU、LCPU 时)
  2^{128} ≦ | 运算结果 | (出错代码： 4141)
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  (通用型 QCPU、LCPU 时) (出错代码： 4140)

程序示例

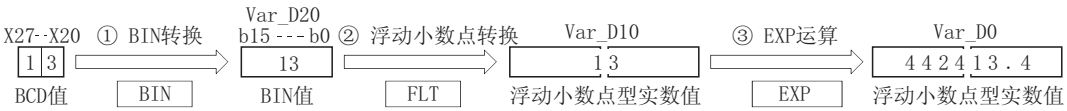
以下为对 X20 ～ X27 中以 BCD 2 位设置的值进行指数运算后，存储到 32 位浮动小数点型实数 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
BIN(X0, K2X20, Var_D20);
OUT(Var_D20>89, M0);

IF NOT M0 THEN
    FLT(TRUE, Var_D20, Var_D10);
    EXP(TRUE, Var_D10, Var_D0);
END_IF;
```

[X20 ～ X27 中指定了 13 时的动作]



*2: 由于 loge 2^{129}=89.4 俗傷，只有 X20 ～ X27 的 BCD 值为 89 以下时，运算结果才会低于 2^{129}。
由于设置为 90 以上的值时将会变为运算出错状态，因此在设置为 90 以上的值时，应将 M0 置为 ON，不执行运算。

要 点

从自然对数至常用对数的转换
在 CPU 中，运算是通过自然对数进行。

求出常用对数中的值的情况下，应指定将 ⓢ 中常用对数的值用 0.43429 相除后的值。

$$10^{x}=e^{\frac{x}{0.43429}}$$

7.12.22 浮动小数点指数运算（双精度）

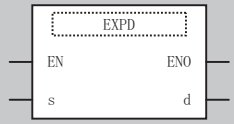
EXPD

Basic High performance Universal L CPU

EXPD(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= EXPD (EN, s, d);
```

□ 中放入下述指令。

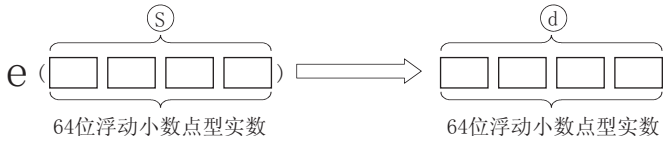
EXPD　　EXPDP

输入自变量，EN:　执行条件　：位
　　s:　进行指数运算的数据　：双精度实数
输出自变量，ENO:　执行结果　：位
　　d:　运算结果　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	-	○		-				○	-
ⓓ	-	○		-				-	-

功 能

(1) 对ⓢ中指定的值进行指数运算后，将运算结果存储到ⓓ中指定的软元件中。
ⓢ中可指定的范围为 0 或 $2^{-1022} \sim 2^{1024}$。



(2) 在指数运算中，将底 (e) 设置为 “2.71828” 进行运算。

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

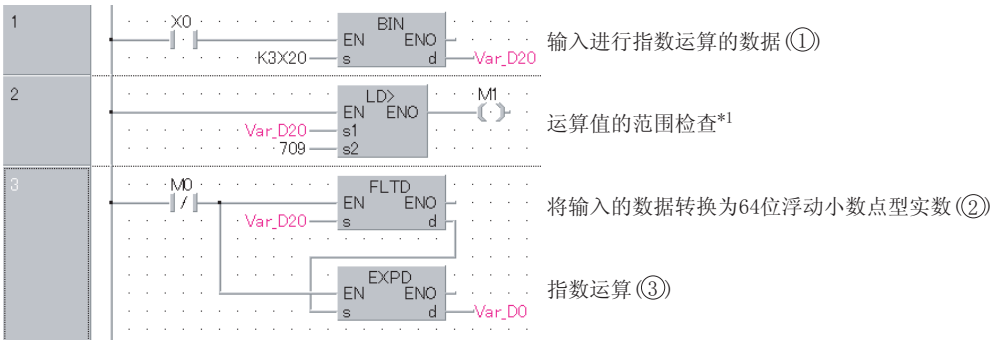
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$（出错代码： 4141）

程序示例

以下为对 X20 ～ X27 中以 BCD 2 位设置的值进行指数运算后，存储到 64 位浮动小数点型实数 Var_D0 中的程序。

[结构体梯形图]




```
[ST]
BIN(X0,K3X20,Var_D20);
OUT(Var_D20>709,M1);
IF NOT(M0) THEN
    FLTD(TRUE,Var_D20,Var_D10);
    EXPD(TRUE,Var_D10,Var_D0);
END_IF;
```

[X20 ～ X27 中指定了 13 时的动作]

*1: 由于 loge 2^{1024}=709.7832，因此只有 X20 ～ X31 的 BCD 值为 709 以下时，运算结果才会低于 2^{1024}。由于设置为 710 以上的值时将会变为运算出错，因此设置为 710 以上的值时，应将 M0 置为 ON，不执行运算。

要 点

从自然对数至常用对数的转换

在 CPU 中，运算是通过自然对数进行。

求出常用对数中的值的情况下，应指定将 ⓢ 中常用对数的值用 0.43429 相除后的值。

$$10^{x}=e^{\frac{x}{0.43429}}$$

7.12.23 浮动小数点自然对数运算（单精度）

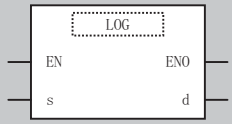
LOG

Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

LOG(P)	P：执行条件　　：↑

结构体梯形图



ST

```
ENO:=  LOG  (EN, s, d);
```

[　　] 中放入下述指令。

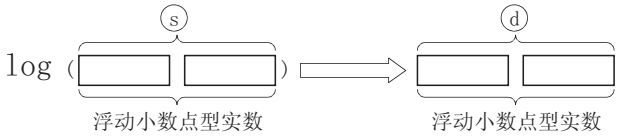
LOG　　LOGP

输入自变量，EN：执行条件　　：位
　　s：进行自然对数运算的数据　　：单精度实数 /ANY32
输出自变量，ENO：执行结果　　：位
　　d：运算结果　　：单精度实数 /ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 E	其它
	位	字		位	字				
ⓢ	–	○		–	○		–	○	–
ⓓ	–	○		–	○		–	–	–

功 能

(1) 进行以ⓢ中指定的值的自然对数 (e) 为底时的对数运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时不能进行运算。）

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

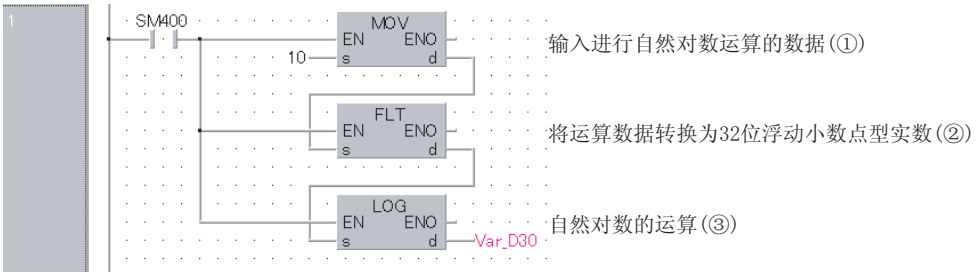
- ⓢ中指定的值为负值时。（出错代码： 4100）
- ⓢ中指定的值为 0 时。（出错代码： 4100）
- 指定软元件的内容超出以下范围时。
  （通用型 QCPU、LCPU 时）
  0，2^{-126} ≦ | 指定软元件的内容 | ＜ 2^{128}（出错代码： 4140）
- 指定软元件的内容为 -0 时。*1
  （基本型 QCPU，高性能型 QCPU 时）（出错代码： 4100）

  *1 ： 有的 CPU 模块即使指定为 -0 也不会变为运算出错状态。
  详细内容请参阅 MELSEC-Q/L/F 结构体编程手册（基础篇）。
- 运算结果超出了下述范围时。（发生了上溢时。）
  （通用型 QCPU、LCPU 时）
  2^{128} ≦ | 运算结果 |（出错代码： 4141）
- 指定软元件的内容为 -0、非正规数、非数、±∞时。
  （通用型 QCPU、LCPU 时）（出错代码： 4140）

程序示例

以下为求出 D50 中设置的 10 的自然对数后，存储到 Var_D30 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     MOV(TRUE,10,Var_D50);
     FLT(TRUE,Var_D50,Var_D40);
     LOG(TRUE,Var_D40,Var_D30);
END_IF;
```

[动作]

7.12.24 浮动小数点自然对数运算（双精度）

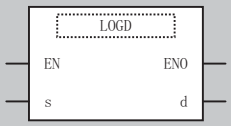
LOGD

Basic High performance Universal L CPU

LOGD(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= LOGD (EN, s, d);
```

□ 中放入下述指令。

LOGD　　LOGDP

输入自变量，EN：　执行条件　　：位

　　s：　进行自然对数运算的数据　　：双精度实数

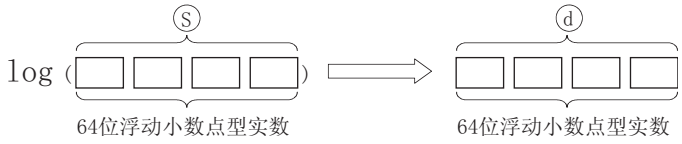
输出自变量，ENO：　执行结果　　：位

　　d：　运算结果　　：双精度实数

设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	－	○		－				○	－
ⓓ	－	○		－				－	－

功 能

(1) 进行以ⓢ中指定的值的自然对数（e）为底时的对数运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时不能进行运算。）

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

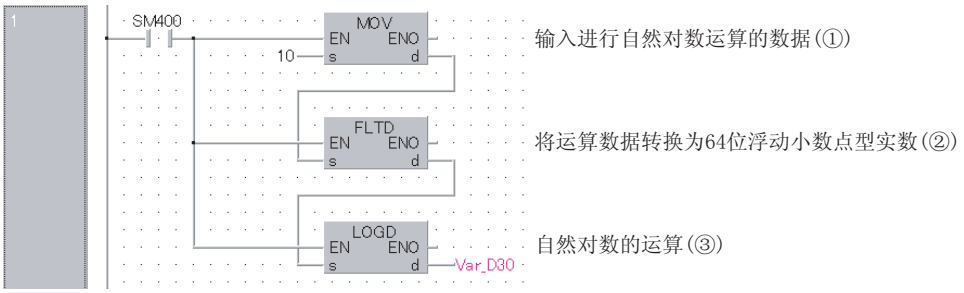
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的值为负值时。（出错代码： 4100）
- ⓢ中指定的值为 0 时。（出错代码： 4100）
- 指定软元件的内容超出以下范围时。（出错代码： 4140）
  0，$2^{-1022} \leqq |$指定软元件的内容$| < 2^{1024}$
- 指定软元件的内容为 -0 时。（出错代码： 4140）
- 运算结果超出了下述范围时。（发生了上溢时。）
  $2^{1024} \leqq |$ 运算结果 $|$（出错代码： 4141）

程序示例

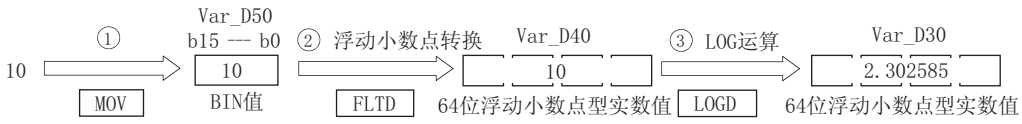
以下为求出 D50 中设置的 “10” 的自然对数后，存储到 Var_D30 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     MOV(TRUE,10,Var_D50);
     FLTD(TRUE,Var_D50,Var_D40);
     LOGD(TRUE,Var_D40,Var_D30);
END_IF;
```

[动作]



7.12.25 浮动小数点常用对数运算（单精度）

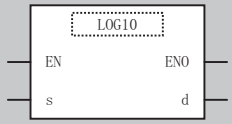
LOG10

Basic　High performance　Ver. Universal　L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

LOG10(P)	P：执行条件　　：

结构体梯形图



ST

```
ENO:= LOG10 (EN, s, d);
```

□ 中放入下述指令。

LOG10　　LOG10P

输入自变量，EN:　执行条件　：位

s:　进行常用对数运算的数据或存储数据的软元件的起始编号　：单精度实数

输出自变量，ENO:　执行结果　：位

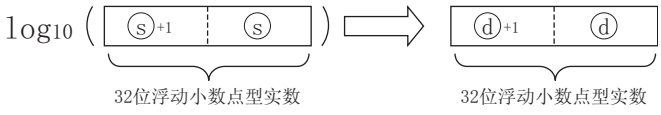
d:　存储运算结果的软元件的起始编号　：单精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	–	○		–	○			△*1	–
ⓓ	–	○		–	○			–	–

*1：只能使用单精度实数。

功 能

(1) 对ⓢ中指定的值进行常用对数（以 10 为底的对数）运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时不能进行运算。）

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

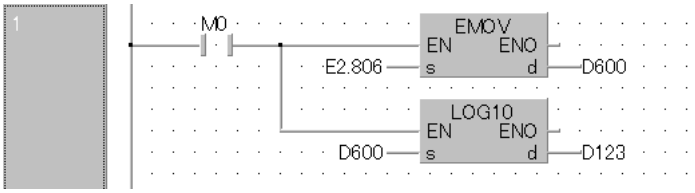
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的值为负值时。 (出错代码： 4100)
- ⓢ 中指定的值为 0 时。 (出错代码： 4100)
- 指定软元件的内容超出以下范围时。 (出错代码： 4140)
  $2^{-126} \leqq |$ 指定软元件的内容 $| < 2^{128}$
- ⓢ 中指定的值为 -0 时。 (出错代码： 4140)
- 运算结果在下述范围内时。(发生了上溢时。)
  $2^{128} \leqq |$ 运算结果 $|$ (出错代码： 4141)

程序示例

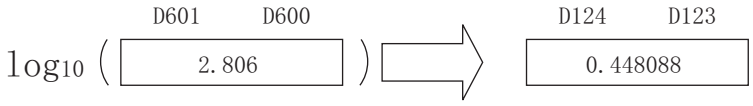
以下为 M0 变为 ON 时，对 D600、D601 中存储的 32 位浮动小数点型实数进行常用对数计算后，将运算结果存储到 D123、D124 中的程序。

[结构体梯形图]




```
[ST]
IF M0 THEN
     EMOV(TRUE,E2.806,D600);
     LOG10(TRUE,D600,D123);
END_IF;
```

[动作]



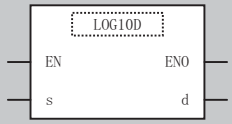
7.12.26 浮动小数点常用对数运算（双精度）

LOG10D

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为“10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为“10102”以后

LOG10D(P)	P：执行条件　　：

结构体梯形图



ST

```
ENO:=  LOG10D  (EN, s, d);
```

[　　] 中放入下述指令。

LOG10D　　LOG10DP

输入自变量，EN：执行条件　：位

s：进行常用对数运算的数据或存储数据的软元件的起始编号　：双精度实数

输出自变量，ENO：执行结果　：位

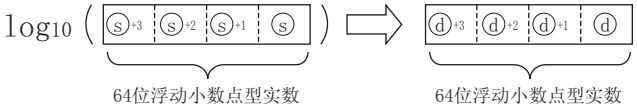
d：存储运算结果的软元件的起始编号　：双精度实数

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			E	
ⓢ	–	○		–	○			△*1	–
ⓓ	–	○		–	○			–	–

*1：只能使用双精度实数。

功 能

(1) 对ⓢ中指定的值进行常用对数（以 10 为底的对数）运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ中指定的值只能设置为正数。（负数时不能进行运算。）

(3) 运算结果为 -0 或发生了下溢时，将运算结果作为 0 处理。

出 错

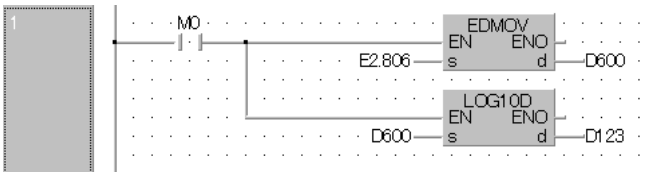
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的值为负值时。 (出错代码： 4100)
- ⓢ中指定的值为 0 时。 (出错代码： 4100)
- 指定软元件的内容超出以下范围时。 (出错代码： 4140)
  0, $2^{-1022} \leqq |$ 指定软元件的内容 $| < 2^{1024}$
- ⓢ中指定的值为 -0 时。 (出错代码： 4140)
- 运算结果在下述范围内时。(发生了上溢时。)
  $2^{1024} \leqq |$ 运算结果 $|$ (出错代码： 4141)

程序示例

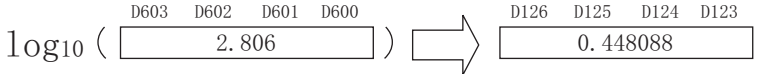
以下为 M0 变为 ON 时，对 D600 ～ D603 中存储的 64 位浮动小数点型实数进行常用对数计算后，将运算结果存储到 D123 ～ D126 中的程序。

[结构体梯形图]




```
[ST]
IF M0 THEN
     EDMOV(TRUE,E2.806,D600);
     LOG10D(TRUE,D600,D123);
END_IF;
```

[动作]



7.12.27 随机数发生、系列变更

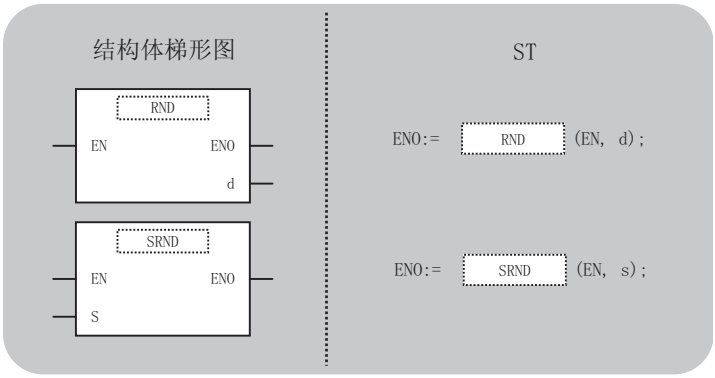
RND, SRND

Ver. Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为“04122”以后

RND(P)
SRND(P)

P：执行条件　　：



中放入下述指令。

RND	RNDP
SRND	SRNDP

输入自变量，EN：　执行条件　：位
　　s：　随机数系列数据或存储随机数系列数据的软元件的起始编号（SRND 时）　：ANY16
输出自变量，ENO：　执行结果　：位
　　d：　存储随机数的软元件的起始编号（RND 的场合）　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○				○	-
ⓓ				○				-	-

功 能

随机数发生指令根据某个计算公式发生随机数。在计算公式中将上次的计算结果作为系数使用。通过系列变更指令可以对随机数的计算模式进行变更。

RND(P)

发生 0 ～ 32767 的随机数后，存储到ⓓ中指定的软元件中。

SRND(P)

根据Ⓢ中指定的软元件中存储的 16 位 BIN 数据的内容，对随机数系列进行变更。

出 错

不存在 RND(P)、SRND(P) 指令相关的运算出错。

程序示例

(1) 以下为将 X10 变为 ON 时，将随机数存储到 Var_D100 中的程序。

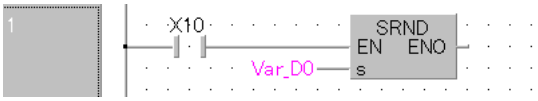
[结构体梯形图]

[ST]

```
RND(X10,Var_D100);
```

(2) 以下为 X10 变为 ON 时，根据 Var_D0 的内容对随机数系列进行变更的程序。

[结构体梯形图]



[ST]

```
SRND(X10,Var_D0);
```

7.12.28 BCD4 位、8 位平方根

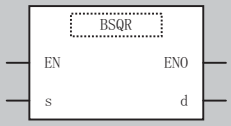
BSQR, BDSQR

Basic | High performance | Universal | L CPU

BSQR(P)
BDSQR(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= BSQR (EN, s, d);

[ ] 中放入下述指令。

BSQR	BSQRP
BDSQR	BDSQRP

输入自变量, EN: 执行条件 : 位
s: 进行平方根运算的数据 : ANY16/32
输出自变量, ENO: 执行结果 : 位
d: 运算结果 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ				○				○	-
ⓓ				○				-	-

功 能

BSQR(P)

(1) 对ⓢ中指定的值进行平方根运算后，将运算结果存储到ⓓ中指定的软元件中。

$$\sqrt{(s)} = (d)\text{整数部分}.(d+1)\text{小数部分}$$

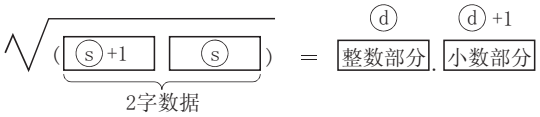
(2) ⓢ中指定值最大为 4 位数的 BCD 值 (0 ～ 9999)。

(3) ⓓ, ⓓ+1 的运算结果分别以 BCD 值在 0 ～ 9999 的范围内被存储。

(4) 运算结果为将小数部分的第 5 位进行四舍五入后的值。
因此，小数部分的第 4 位将产生± 1 的误差。

BDSQR(P)

(1) 对ⓢ，ⓢ+1 中指定的值进行平方根运算后，将运算结果存储到ⓓ中指定的软元件中。



(2) ⓢ，ⓢ+1 中指定的值最大为 8 位数的 BCD 值 (0 ～ 99999999)。

(3) ⓓ，ⓓ+1 的运算结果分别以 BCD 值在 0 ～ 9999 的范围内被存储。

(4) 运算结果为将小数部分的第 5 位进行四舍五入后的值。
因此，小数部分的第 4 位将产生± 1 的误差。

出 错

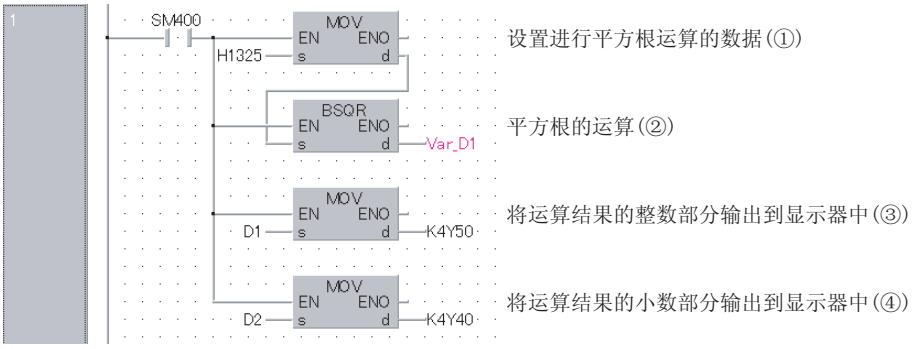
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的数据不是 BCD 值时。 (出错代码： 4100)

程序示例

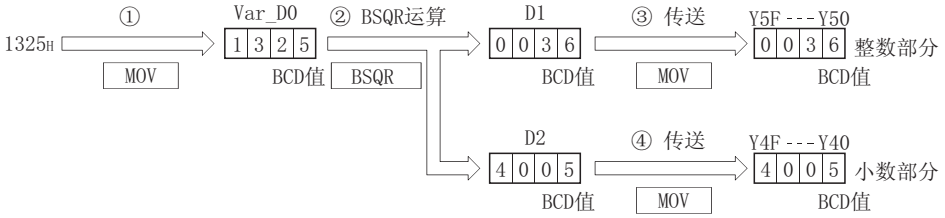
(1) 以下为对 BCD 值 1325 进行平方根运算后，将整数部分以 BCD 4 位输出到 Y50 ～ Y5F 中，将小数部分以 BCD 4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]



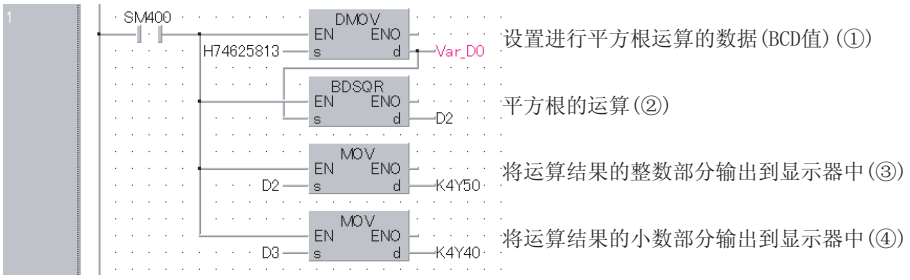

```
[ST]
IF SM400 THEN
     MOV(TRUE,H1325,Var_D0);
     BSQR(TRUE,Var_D0,Var_D1);
     MOV(TRUE,D1,K4Y50);
     MOV(TRUE,D2,K4Y40);
END_IF;
```

[动作]



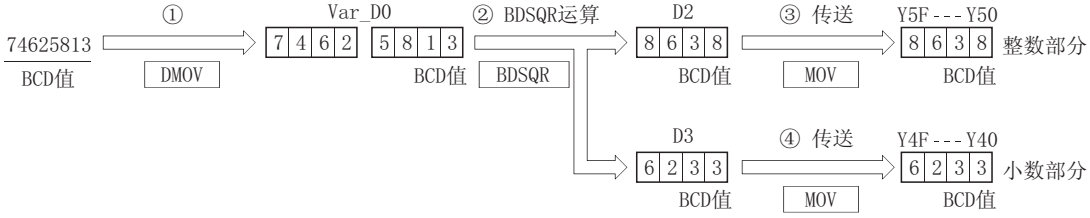
(2) 以下为对 BCD 值 74625813 进行平方根运算后，将整数部分以 BCD 4 位输出到 Y50 ～ Y5F 中，将小数部分以 BCD 4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
IF SM400 THEN
     DMOV(TRUE,H74625813,Var_D0);
     BDSQR(TRUE,Var_D0,D2);
     MOV(TRUE,D2,K4Y50);
     MOV(TRUE,D3,K4Y40);
END_IF;
```

[动作]



7.12.29 BCD 型 SIN 运算

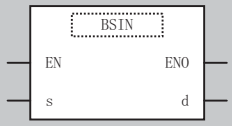
BSIN

Basic | High performance | Universal | L CPU

BSIN(P)

P: 执行条件　　: ↑

结构体梯形图



ST

```
ENO:= BSIN (EN, s, d);
```

[　　] 中放入下述指令。

BSIN　　BSINP

输入自变量，EN:　执行条件　: 位

s:　进行 SIN(正弦)运算的数据或存储数据的软元件编号　: ANY16

输出自变量，ENO:　执行结果　: 位

d:　存储运算结果的软元件的起始编号　: ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○						○	-
ⓓ	-	○						-	-

功 能

(1) 对ⓢ中指定的值(角度)进行 SIN(正弦)值运算后，将运算结果的符号存储到ⓓ中指定的软元件中，将运算结果存储到ⓓ+1，ⓓ+2 中指定的软元件中。

	ⓓ	ⓓ+1	ⓓ+2
SIN ⓢ =	符号	整数部分	. 小数部分

(2) ⓢ中指定的值是在 0 ～ 360°(DEG. 单位)的范围内以 BCD 值进行设置。

(3) 对于ⓓ中存储的运算结果的符号，运算结果为正时存储 0，为负时存储 1。

(4) ⓓ+1，ⓓ+2 中存储的运算结果是 BCD 值，范围为 -1.000 ～ 1.000。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

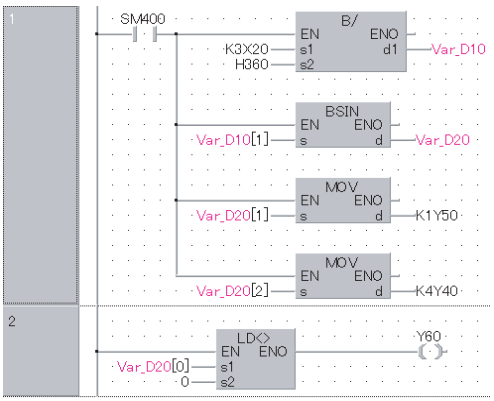
- ⓢ中指定的数据不是 BCD 值时。（出错代码： 4100）
- ⓢ中指定的数据超出了 0 ～ 360 的范围时。（出错代码： 4100）
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为对 X20 ～ X2B 中以 BCD 3 位指定的数据进行 SIN 运算后，将整数部分以 BCD1 位输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

运算结果为负时将 Y60 置为 ON。（X20 ～ X2F 中设置了 360 以上的值的情况下，将设置的值置为 0 ～ 360 的范围内。）

［结构体梯形图］



将输入的角度进行处理使之处于360度的范围内(①)

SIN值的运算(②)

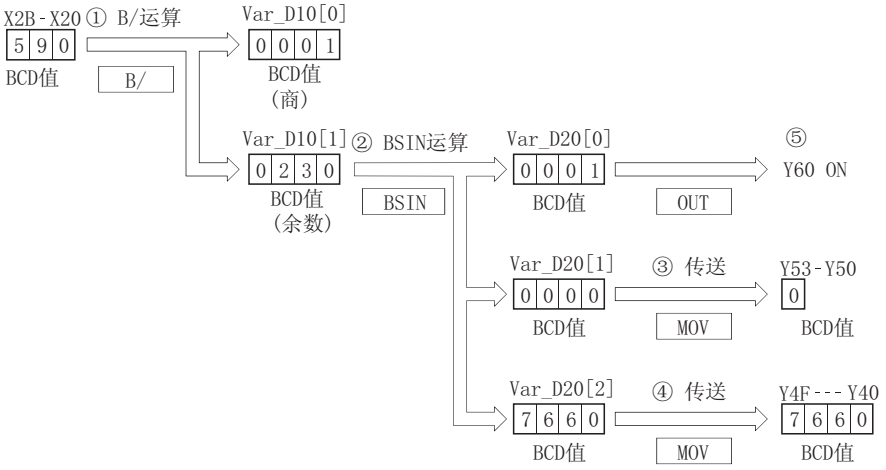
将运算结果的整数部分输出到显示器中(③)

将运算结果的小数部分输出到显示器中(④)

将运算结果的符号置为ON/OFF后输出(⑤)

```
[ST]
IF SM400 THEN
     BINP(TRUE,K3X20,Var_D50);
     BINP(TRUE,H360,Var_D51);
     BCDP(TRUE,Var_D50 MOD Var_D51,Var_D10);
     BSIN(TRUE,Var_D10,Var_D20);
     MOV(TRUE,Var_D20[1],K1Y50);
     MOV(TRUE,Var_D20[2],K4Y40);
END_IF;
OUT(Var_D20[0]<>0,Y60);
```

[X20 ～ X2B 中指定了 590 时的动作]



7.12.30 BCD 型 COS 运算

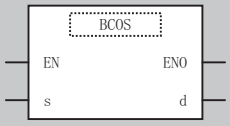
BCOS

Basic | High performance | Universal | L CPU

BCOS(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= BCOS (EN, s, d);
```

□ 中放入下述指令。

BCOS　　BCOSP

输入自变量，EN：　执行条件　：位

　　s：　进行 COS(余弦)运算的数据或存储数据的软元件的起始编号　：ANY16

输出自变量，ENO：　执行结果　：位

　　d：　存储运算结果的软元件的起始编号　：ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
ⓓ	-	○		-					-

功 能

(1) 对ⓢ中指定的值(角度)进行 COS(余弦)值运算后，将运算结果的符号存储到ⓓ中指定的软元件中，将运算结果存储到ⓓ+1，ⓓ+2 中指定的字软元件中。

	ⓓ	ⓓ+1	ⓓ+2
COS ⓢ =	符号	整数部分	. 小数部分

(2) ⓢ中指定的值是在 0 ～ 360°(DEG. 单位)的范围内以 BCD 值进行设置。

(3) 对于ⓓ中存储的运算结果的符号，运算结果为正时存储 0，为负时存储 1。

(4) ⓓ+1，ⓓ+2 中存储的运算结果是 BCD 值，范围为 -1.000 ～ 1.000。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

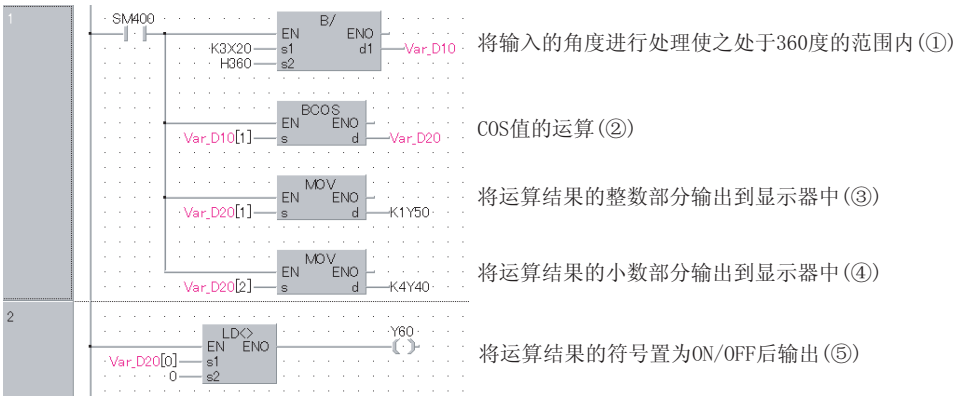
- ⓢ中指定的数据不是 BCD 值时。（出错代码： 4100）
- ⓢ中指定的数据超出了 0 ～ 360 的范围时。（出错代码： 4100）
- ⓢ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为对 X20 ～ X2B 中以 BCD 3 位指定的数据进行 COS 运算后，将整数部分以 BCD1 位输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

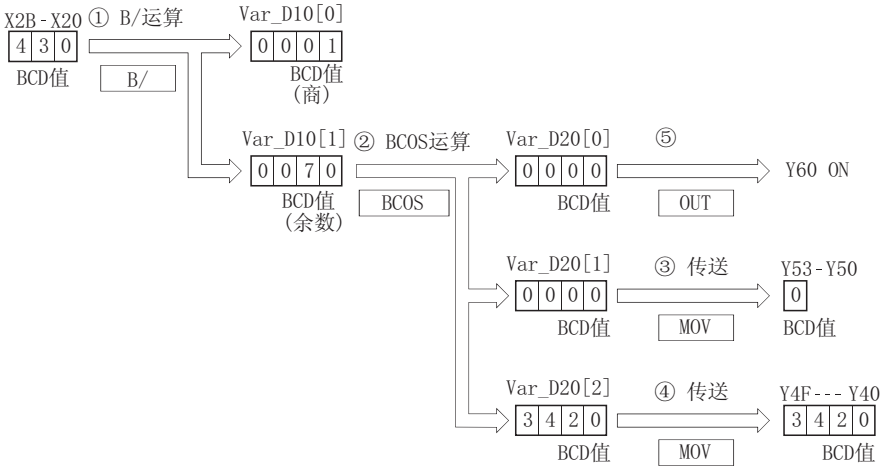
运算结果为负时将 Y60 置为 ON。

［结构体梯形图］



```
[ST]
IF SM400 THEN
     BINP(TRUE,K3X20,Var_D50);
     BINP(TRUE,H360,Var_D51);
     BCDP(TRUE,Var_D50 MOD Var_D51,Var_D11);
     BCOS(TRUE,Var_D11,Var_D20);
     MOV(TRUE,Var_D20[1],K1Y50);
     MOV(TRUE,Var_D20[2],K4Y40);
END_IF;
OUT(Var_D20[0]<>0,Y60);
```

[X20 ～ X2B 中指定了 430 时的动作]



7.12.31 BCD 型 TAN 运算

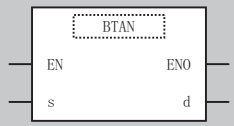
BTAN

Basic | High performance | Universal | L CPU

BTAN(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= BTAN (EN, s, d);
```

[ ] 中放入下述指令。

BTAN BTANP

输入自变量， EN: 执行条件 : 位

s: 进行 TAN(正弦)运算的数据或存储数据的软元件的起始编号 : ANY16

输出自变量， ENO: 执行结果 : 位

d: 存储运算结果的软元件的起始编号 : ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○					-
ⓓ	-	○		-					-

功 能

(1) 对ⓢ中指定的值(角度)进行 TAN(正切)值运算后，将运算结果的符号存储到ⓓ中指定的软元件中，将运算结果存储到ⓓ+1, ⓓ+2 中指定的软元件中。

TAN ⓢ = ⓓ 符号 | ⓓ+1 整数部分 . | ⓓ+2 小数部分

(2) ⓢ中指定的值是在 0 ～ 360°(DEG. 单位)的范围内以 BCD 值进行设置。

(3) 对于ⓓ中存储的运算结果的符号，运算结果为正时存储 0，为负时存储 1。

(4) ⓓ+1, ⓓ+2 中存储的运算结果是 BCD 值，范围为 -57.2900 ～ 57.2900。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

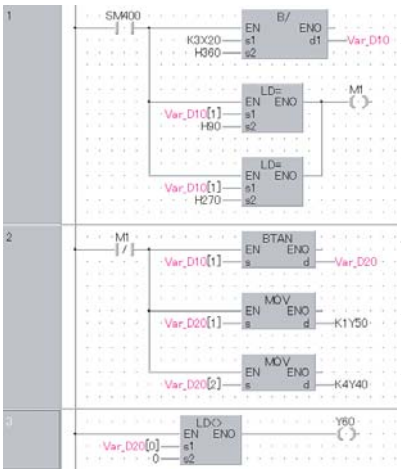
- Ⓢ中指定的数据不是 BCD 值时。（出错代码： 4100）
- Ⓢ中指定的数据超出了 0 ～ 360 的范围时。（出错代码： 4100）
- Ⓢ中指定的数据为 90°、270° 时。（出错代码： 4100）
- Ⓢ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为对 X20 ～ X2B 中以 BCD 3 位指定的数据进行 TAN 运算后将整数部分以 BCD4 位输出到 Y50 ～ Y53 中，将小数部分以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

运算结果为负时将 Y60 置为 ON。

［结构体梯形图］



将输入的角度进行处理使之处于360度的范围内(①)

使用M1作为互锁使输入的角度为90度、270度时不执行处理。

TAN值的运算(②)

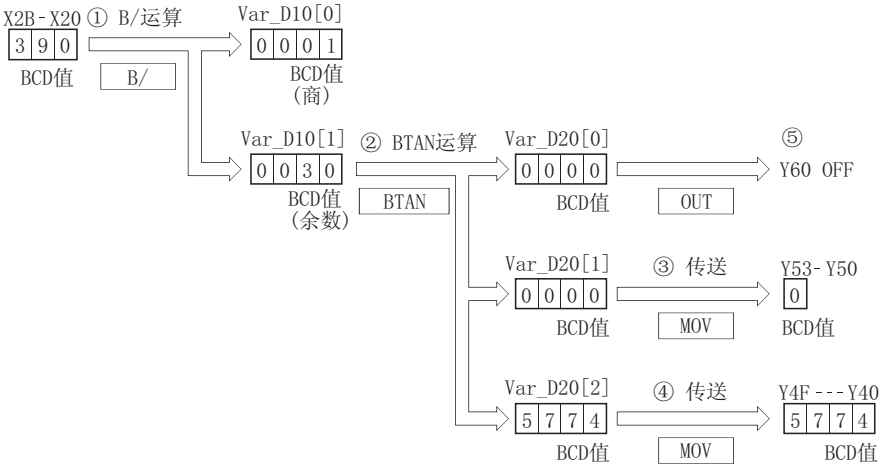
将运算结果的整数部分输出到显示器中(③)

将运算结果的小数部分输出到显示器中(④)

将运算结果的符号置为ON/OFF后输出(⑤)

```
[ST]
IF SM400 THEN
     BINP(TRUE,K3X20,Var_D50);
     BINP(TRUE,H360,Var_D51);
     BCDP(TRUE,Var_D50 MOD Var_D51,Var_D10);
     OUT(Var_D10=H90 OR Var_D10=H270,M1);
END_IF;

IF NOT(M1) THEN
     BTAN(TRUE,Var_D10,Var_D20);
     MOV(TRUE,Var_D20[1],K1Y50);
     MOV(TRUE,Var_D20[2],K4Y40);
END_IF;
OUT(Var_D20[0]<>0,Y60);
```

[X20 ～ X2B 中指定了 390 时的动作]



7.12.32 BCD 型 SIN^{-1} 运算

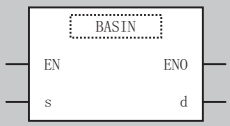
BASIN

Basic High performance Universal L CPU

BASIN(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= BASIN (EN, s, d);

[ ] 中放入下述指令。

BASIN BASINP

输入自变量， EN: 执行条件 : 位
s: 进行 SIN^{-1}（反正弦）运算的数据 : ANY16 的数组 (0..2)
输出自变量， ENO: 执行结果 : 位
d: 存储运算结果的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	-	○		-				-	
ⓓ	○	○		○				-	

功 能

(1) 对ⓢ中指定的值进行 SIN^{-1}（反正弦）值运算后，将运算结果（角度）存储到ⓓ中指定的软元件中。

SIN^{-1} (ⓢ 符号 | ⓢ+1 整数部分 . | ⓢ+2 小数部分) = ⓓ

(2) 在ⓢ中设置运算数据的符号。
运算结果为正时存储 0，为负时存储 1。

(3) ⓢ+1，ⓢ+2 中分别以 BCD 值存储运算数据的整数部分及小数部分。
（可设置范围为 0 ～ 1.0000。）

(4) ⓓ中存储的运算结果为 BCD 值，其范围为 0 ～ 90°、270 ～ 360°（DEG. 单位）。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

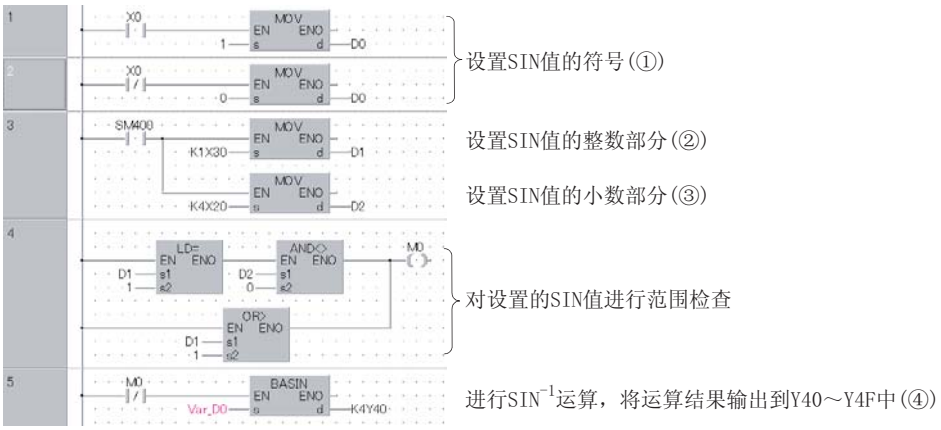
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的数据不是 BCD 值时。 (出错代码： 4100)
- ⓢ 中指定的数据超出了 -1.000 ～ 1.000 的范围时。 (出错代码： 4100)
- ⓢ 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

程序示例

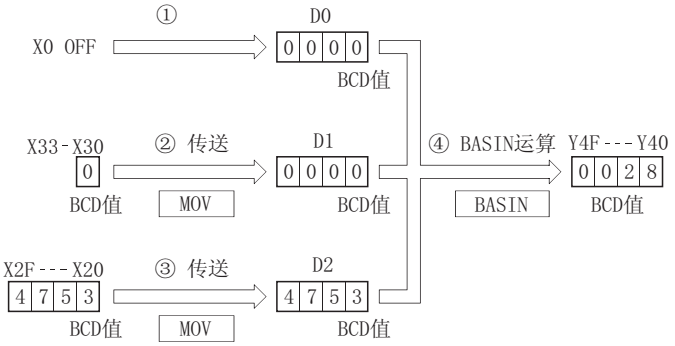
以下为对将 X0 作为正负符号 (OFF 时正，ON 时负)、将 X30 ～ X33 作为 BCD 1 位的整数部分、将 X20 ～ X2F 作为 BCD4 位的小数部分的值进行 SIN^{-1} 运算后，将求出的角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]



```
[ST]
MOV(X0,1,D0);
MOV(NOT(X0),0,D0);
IF SM400 THEN
     MOV(TRUE,K1X30,D1);
     MOV(TRUE,K4X20,D2);
END_IF;
OUT((D1=1 AND D2<>0) OR D1>1,M0);
BASIN(NOT(M0),Var_D0,K4Y40);
```

[X20 ～ X33 中指定了 0.4753 时的动作]



7.12.33 BCD 型 COS $^{-1}$ 运算

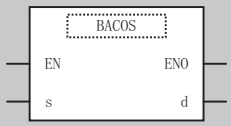
BACOS

Basic | High performance | Universal | L CPU

BACOS(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BACOS (EN, s, d);

□ 中放入下述指令。

BACOS　　BACOSP

输入自变量，EN:　执行条件　：位

　　s:　进行 COS^{-1}（反余弦）运算的数据　：ANY16 的数组（0..2）

输出自变量，ENO:　执行结果　：位

　　d:　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	－	○		－				－	－
ⓓ	○	○		○				－	－

功 能

(1) 对ⓢ中指定的值进行 COS^{-1}（反余弦）值运算后，将运算结果（角度）存储到ⓓ中指定的软元件中。

ⓢ　ⓢ+1　ⓢ+2

COS^{-1}（符号　整数部分.小数部分）＝ⓓ

(2) 在ⓢ中设置运算数据的符号。

运算结果为正时存储 0，为负时存储 1。

(3) ⓢ+1，ⓢ+2 中分别以 BCD 值存储运算数据的整数部分及小数部分。

（可设置范围为 0 ～ 1.0000。）

(4) ⓓ中存储的运算结果为 BCD 值，其范围为 0 ～ 180°（DEG. 单位）。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

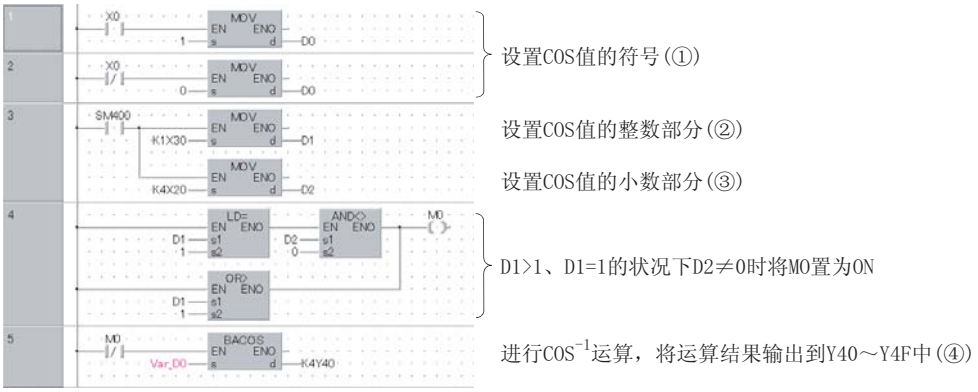
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的数据不是 BCD 值时。（出错代码： 4100）
- ⓢ中指定的数据超出了 -1.000 ～ 1.000 的范围时。（出错代码： 4100）
- ⓢ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

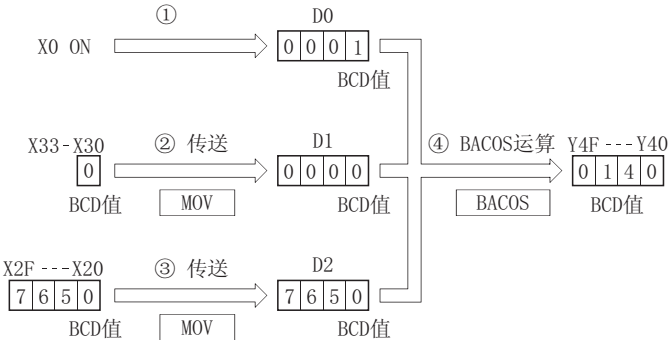
以下为对将 X0 作为正负符号（OFF 时正，ON 时负）、将 X30 ～ X33 作为 BCD 1 位的整数部分、将 X20 ～ X2F 作为 BCD4 位的小数部分的值进行 COS^{-1} 运算后，将求出的角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

［结构体梯形图］



```
[ST]
MOV(X0,1,D0);
MOV(NOT(X0),0,D0);
IF SM400 THEN
     MOV(TRUE,K1X30,D1);
     MOV(TRUE,K4X20,D2);
END_IF;
OUT((D1=1 AND D2<>0) OR D1>1,M0);
BACOS(NOT(M0),Var_D0,K4Y40);
```

[X0,X20 ～ X33 中指定了 -0.7650 时的动作]



7.12.34 BCD 型 TAN^{-1} 运算

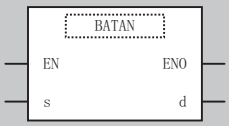
BATAN

Basic | High performance | Universal | L CPU

BATAN(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= BATAN (EN, s, d);

□ 中放入下述指令。

BATAN　　BATANP

输入自变量，EN:　执行条件　：位
　　s:　进行 TAN^{-1}（反正切）运算的数据　：ANY16 的数组 (0..2)
输出自变量，ENO:　执行结果　：位
　　d:　存储运算结果的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	－	○		－				－	－
ⓓ	○	○		○				－	－

功 能

(1) 对ⓢ中指定的值进行 TAN^{-1}（反正切）值运算后，将运算结果（角度）存储到ⓓ中指定的软元件中。

TAN^{-1}（ ⓢ 符号 | ⓢ+1 整数部分 . | ⓢ+2 小数部分 ）＝ⓓ

(2) 在ⓢ中设置运算数据的符号
运算结果为正时存储 0，为负时存储 1。

(3) ⓢ+1，ⓢ+2 中分别以 BCD 值存储运算数据的整数部分及小数部分。
（可设置范围为 0 ～ 9999.9999。）

(4) ⓓ中存储的运算结果为 BCD 值，其范围为 0 ～ 90°、270 ～ 360°（DEG. 单位）。

(5) 运算结果为将小数部分的第 5 位进行四舍五入后的值。

出 错

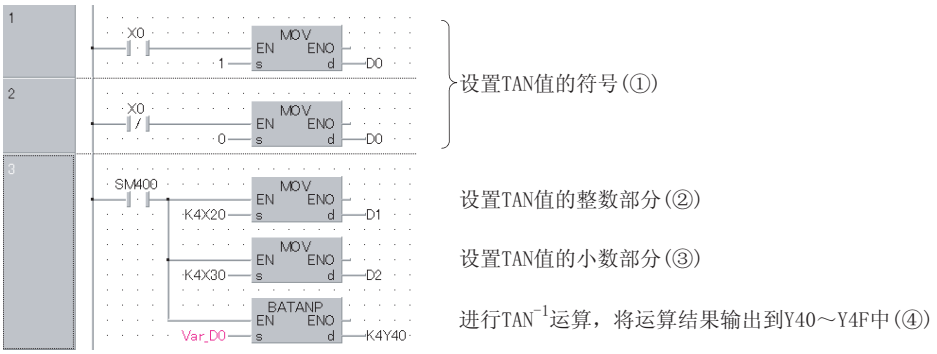
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的数据不是 BCD 值时。 （出错代码： 4100）
- ⓢ 中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

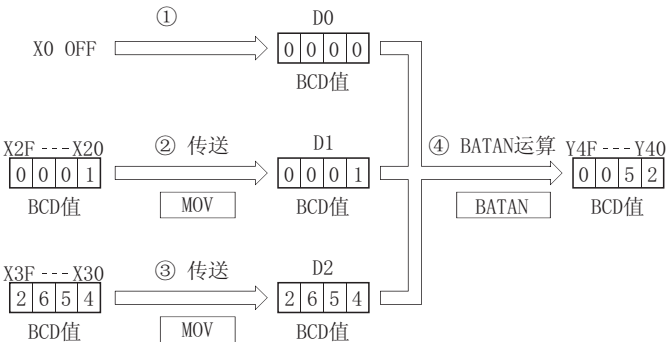
以下为对将 X0 作为正负符号 (OFF 时正，ON 时负)、将 X20 ～ X2F 作为 BCD4 位的整数部分、将 X30 ～ X3F 作为 BCD4 位的小数部分的值进行 TAN^{-1} 运算后，将求出的角度以 BCD4 位输出到 Y40 ～ Y4F 中的程序。

[结构体梯形图]




```
[ST]
MOV(X0,1,D0);
MOV(NOT(X0),0,D0);
IF SM400 THEN
     MOV(TRUE,K4X20,D1);
     MOV(TRUE,K4X30,D2);
     BATANP(TRUE,Var_D0,K4Y40);
END_IF;
```

[X0,X20 ～ X2F 中指定了 1.2654 时的动作]



7.13 数据控制指令

7.13.1 BIN16 位 /32 位上下限极限控制

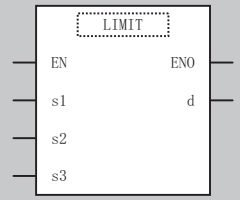
LIMIT, DLIMT

Basic | High performance | Universal | L CPU

LIMIT(P)
DLIMT(P)

P: 执行条件 : ↵

结构体梯形图



ST

ENO:= LIMIT (EN, s1, s2, s3, d);

[ ] 中放入下述指令。

LIMIT	LIMITP
DLIMIT	DLIMITP

输入自变量， EN: 执行条件 : 位
s1: 下段极限值 : ANY16/32
s2: 上段极限值 : ANY16/32
s3: 根据上下段极限控制进行控制的输入值 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 根据上下段极限控制进行了控制的输出值 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1				○				○	-
⑤2				○				○	-
⑤3				○				○	-
ⓓ				○				-	-

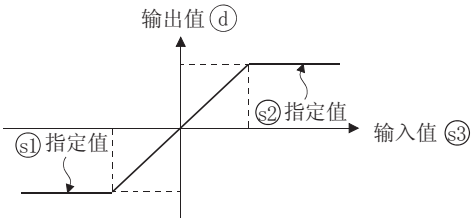
功 能

LIMIT(P)

(1) 通过③中指定的输入值 (BIN16 位值)，根据①，②中指定的上下限极限值的范围，对ⓓ中指定的软元件中存储的输出值进行控制。

对输出值按下述方法进行控制。

- ① 下限值 ＞ ③ 输入值 时 ..... ① 下限值 → ⓓ 输出值
- ② 上限值 ＜ ③ 输入值 时 ..... ② 上限值 → ⓓ 输出值
- ① 下限值 ≦ ③ 输入值 ≦ ② 上限值 时..... ③ 输入值 → ⓓ 输出值



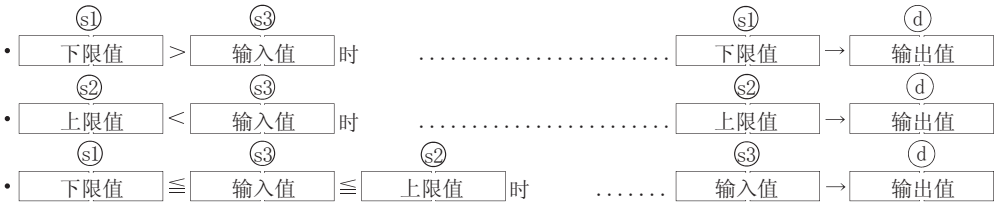
(2) ①，②，③中可指定的值的范围为 -32768 ～ 32767。

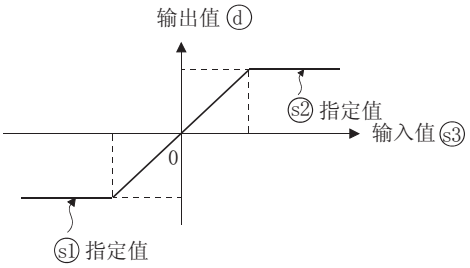
(3) 仅根据上限极限值进行控制的情况下，在①中指定的下限极限值中设置 -32768。

(4) 仅根据下限极限值进行控制的情况下，在②中指定的上限极限值中设置 32767。

DLIMIT(P)

(1) 通过③中指定的输入值 (BIN32 位值)，根据①，②中指定的上下限极限值的范围，对ⓓ中指定的软元件中存储的输出值进行控制。





(2) ①，②，③中可指定的值的范围为 -2147483648 ～ 2147483647。

(3) 仅根据上限极限值进行控制的情况下，在①中指定的下限极限值中设置 -2147483648。

(4) 仅根据下限极限值进行控制的情况下，在②中指定的上限极限值中设置 2147483647。

出 错

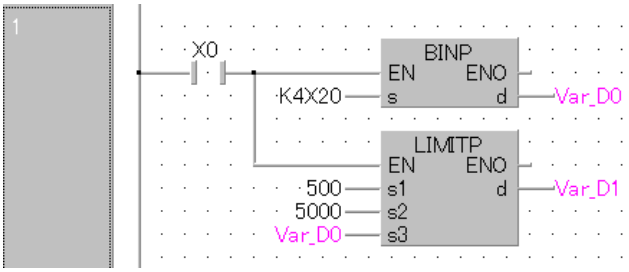
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ㉛中指定的下限极限值大于㉜中指定的上限极限值时。　　（出错代码： 4100）

程序示例

(1) 以下为 X0 变为 ON 时，对 X20 ～ X2F 中以 BCD 值设置的数据进行 500 ～ 5000 的极限控制后，存储到 Var_D1 中的程序。

[结构体梯形图]



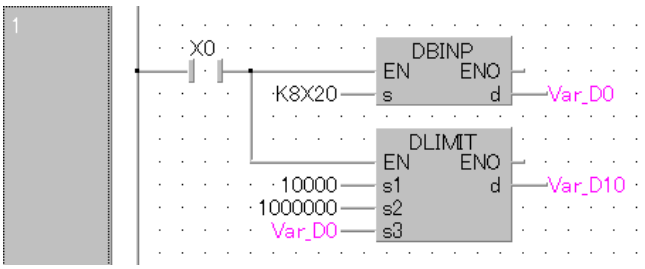

```
[ST]
IF X0 THEN
    BINP(TRUE,K4X20,Var_D0);
    LIMITP(TRUE,500,5000,Var_D0,Var_D1);
END_IF;
```

[动作]

- Var_D0＜500 的情况下，Var_D1 变为 500。
  例　Var_D0=400→Var_D1=500
- 500≦Var_D0≦5000 的情况下，Var_D1 变为 Var_D0 的值。
  例　Var_D0=1300→Var_D1=1300
- 5000＜Var_D0 的情况下，Var_D1 变为 5000。
  例　Var_D0=9600→Var_D1=5000

(2) 以下为 X0 变为 ON 时，对 X20 ～ X3F 中以 BCD 值设置的数据进行 10000 ～ 1000000 的极限控制后，将结果存储到 Var_D10 中的程序。

[结构体梯形图]




```
[ST]
IF X0 THEN
    DBINP(TRUE,K8X20,Var_D0);
    DLIMIT(TRUE,10000,1000000,Var_D0,Var_D10);
END_IF;
```

[动作]

- Var_D0＜10000 的情况下，Var_D10 变为 10000。
  例　Var_D0=400 → Var_D10=10000
- 10000≦Var_D0≦1000000 的情况下，Var_D10 变为 Var_D0 的值。
  例　Var_D0=345678 → Var_D10=345678
- 1000000＜Var_D0 的情况下，Var_D10 变为 1000000。
  例　Var_D0=9876543 → Var_D10=1000000

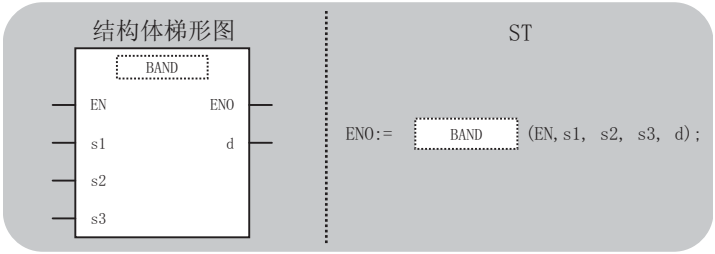
7.13.2 BIN16 位 /32 位死区控制

BAND, DBAND

Basic | High performance | Universal | L CPU

BAND(P)
DBAND(P)

P: 执行条件 : 



[ ] 中放入下述指令。

BAND	BANDP
DBAND	DBANDP

输入自变量，EN: 执行条件 : 位
s1: 死区（无输出区域）的下限值 : ANY16/32
s2: 死区（无输出区域）的上限值 : ANY16/32
s3: 通过死区控制进行控制的输入值 : ANY16/32
输出自变量，ENO: 执行结果 : 位
d: 通过死区控制进行了控制的输出值 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1				○				○	-
⑤2				○				○	-
⑤3				○				○	-
ⓓ				○				-	-

7 应用指令

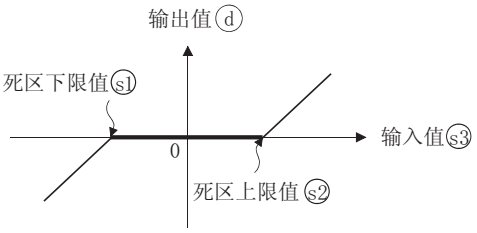
功 能

BAND(P)

(1) 通过㉓中指定的输入值 (BIN16 位值)，根据㉑， ㉒中指定的死区的上下限范围，对ⓓ中指定的软元件中存储的输出值进行控制。

对输出值按下述方法进行控制。

- ㉑ 下限值 ＞ ㉓ 输入值 时 ..... ㉓ 输入值 － ㉑ 下限值 → ⓓ 输出值
- ㉒ 上限值 ＜ ㉓ 输入值 时 ..... ㉓ 输入值 － ㉒ 上限值 → ⓓ 输出值
- ㉑ 下限值 ≦ ㉓ 输入值 ≦ ㉒ 上限值 时 0 → ⓓ 输出值



(2) ㉑， ㉒， ㉓中可指定的值的范围为 -32768 ～ 32767。

(3) ⓓ中存储的输出值是带符号的 16 位 BIN 值。因此，运算结果超出了 -32768 ～ 32767 的情况如下所示。

死区下限值㉑ 10
输入值㉓ ..... -32768
} 时

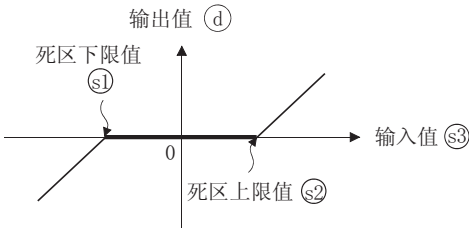
输出值 =-32768-10=8000H-AH=7FF6H=32758。

DBAND(P)

(1) 通过㉓中指定的输入值 (BIN32 位值)，根据㉑， ㉒中指定的死区的上下限范围，对ⓓ中指定的软元件中存储的输出值进行控制。

对输出值按下述方法进行控制。

- ⓢ1 下限值 ＞ ⓢ3 输入值 时 ····· ⓢ3 输入值 － ⓢ1 下限值 → ⓓ 输入值
- ⓢ2 上限值 ＜ ⓢ3 输入值 时 ····· ⓢ3 输入值 － ⓢ2 上限值 → ⓓ 输入值
- ⓢ1 下限值 ≦ ⓢ3 输入值 ≦ ⓢ2 上限值 时 ····· 0 → ⓓ 输入值



(2) ㉑， ㉒， ㉓中可存储的值的范围为 -2147483648 ～ 2147483647。

(3) ④中存储的输出值为带符号的 32 位 BIN 值。因此，运算结果超出了 -2147483648 ～ 2147483647 的情况如下所示。

死区下限值① 1000
输入值③ ..... -2147483648
} 时

输出值 =-2147483648-1000=80000000H-000003E8H
=7FFFFC18H=2147482648。

出 错

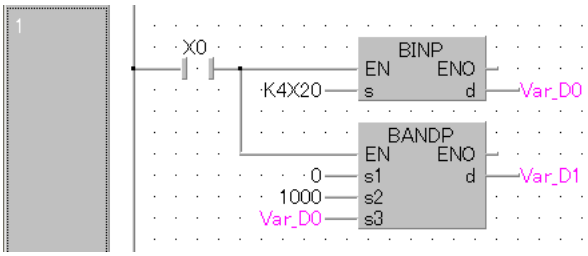
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ①中指定的下限值大于②中指定的上限值时。 (出错代码： 4100)

程序示例

(1) 以下为 X0 变为 ON 时，对 X20 ～ X2F 中以 BCD 值设置的数据进行 0 ～ 1000 的死区控制后，存储到 Var_D1 中的程序。

[结构体梯形图]



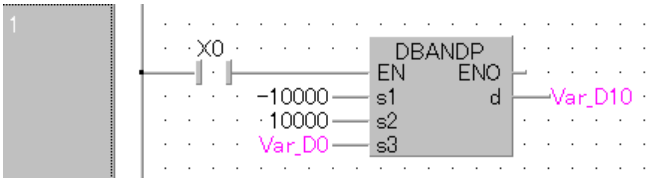

```
[ST]
IF X0 THEN
    BINP(TRUE,K4X20,Var_D0);
    BANDP(TRUE,0,1000,Var_D0,Var_D1);
END_IF;
```

[动作]

- 0≦Var_D0≦1000 的情况下，Var_D1 中将被存储 0。
  例 Var_D0=500→Var_D1=0
- 1000＜Var_D0 的情况下，Var_D1 中将被存储 (Var_D0)-1000 的值。
  例 Var_D0=7000→Var_D1=6000

(2) 以下为 X0 变为 ON 时，对 Var_D0 中设置的数据进行 -10000 ～ 10000 的死区控制后，将结果存储到 Var_D10 中的程序。

[结构体梯形图]



[ST]

```
DBANDP(X0,-10000,10000,Var_D0,Var_D10);
```

[动作]

- Var_D0＜-10000 的情况下，Var_D10 中将存储 Var_D0-(-10000) 的值。
  例　Var_D0=-12345 → Var_D10=-2345
- -10000≦Var_D0≦10000 的情况下，Var_D10 中将存储 0。
  例　Var_D0=6789 → Var_D10=0
- 10000＜Var_D0 的情况下，Var_D10 中将存储 Var_D0-10000 的值。
  例　Var_D0=50000 → Var_D10=40000

7.13.3 BIN16 位 /32 位区域控制

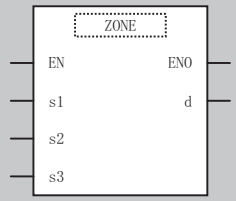
ZONE, DZONE

Basic | High performance | Universal | L CPU

ZONE(P)
DZONE(P)

P: 执行条件 : ↵

结构体梯形图



ST

ENO:= ZONE (EN, s1, s2, s3, d);

□ 中放入下述指令。

ZONE	ZONEP
DZONE	DZONEP

输入自变量， EN: 执行条件 : 位
s1: 对输入值进行加法运算的负的偏置值 : ANY16/32
s2: 对输入值进行加法运算的正的偏置值 : ANY16/32
s3: 用于进行区域控制的输入值 : ANY16/32
输出自变量， ENO: 执行结果 : 位
d: 通过区域控制进行了控制的输出值 : ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
⑤1				○				○	-
⑤2				○				○	-
⑤3				○				○	-
ⓓ				○				-	-

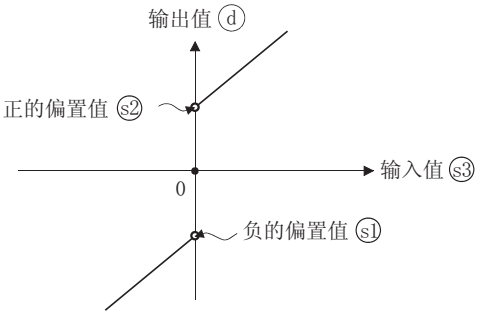
7 应用指令

功 能

ZONE(P)

(1) 对 ⓢ3 中指定的输入值附加 ⓢ1 或 ⓢ2 中指定的偏置值后，存储到 ⓓ 中指定的软元件编号中。
偏置值按以下方式执行。

- ⓢ3 [输入值] < 0 时 ..... ⓢ3 [输入值] + ⓢ1 [负的偏置值] → ⓓ [输出值]
- ⓢ3 [输入值] = 0 时 ..... 0 → ⓓ [输出值]
- ⓢ3 [输入值] > 0 时 ..... ⓢ3 [输入值] + ⓢ1 [正的偏置值] → ⓓ [输出值]



(2) ⓢ1, ⓢ2, ⓢ3 中可指定的值的范围为 -32768 ～ 32767。

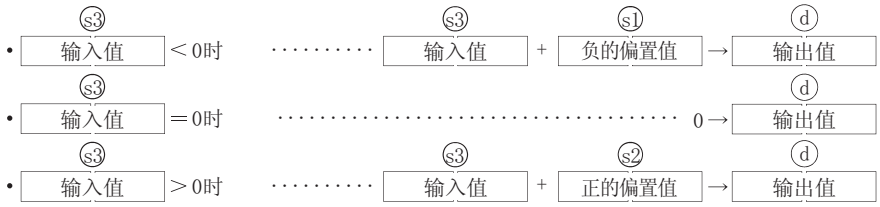
(3) ⓓ 中存储的输出值为带符号的 16 位 BIN 值。因此，运算结果超出了 -32768 ～ 32767 的情况如下所示。

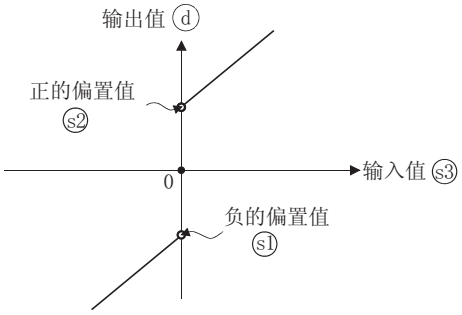
负的偏置值 ⓢ1 ..... -100
输入值 ⓢ3 ..... -32768
} 时

输出值 =-32768+(-100)=8000H+FF9C=7F9CH=32668。

DZONE(P)

(1) 对 ③ 中指定的输入值附加 ① 或 ② 中指定的偏置值后，存储到 ⓓ 中指定的软元件编号中。
偏置值按以下方式执行。





(2) ①， ②， ③ 中可指定的值的范围为 -2147483648 ～ 2147483647。

(3) ⓓ 中存储的输出值为带符号的 32 位 BIN 值。
因此，超出了 -2147483648 ～ 2147483647 的情况如下所示。

负的偏置值① ..... -1000
输入值③ ..... -2147483648
} 时

输出值 =-2147483648+(-1000)=80000000H+FFFFFC18H
=7FFFFC18=2147482648。

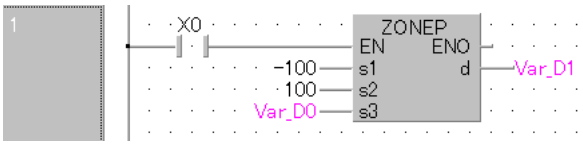
出 错

不存在 ZONE(P)、DZONE(P) 指令相关的运算出错。

程序示例

(1) 以下为 X0 变为 ON 时，以 D0 中设置的数据进行 -100 ～ 100 的区域控制后，存储到 D1 中的程序。

[结构体梯形图]



[ST]

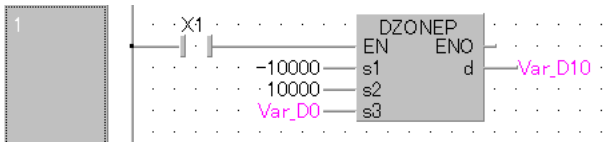
```
ZONEP(X0,-100,100,Var_D0,Var_D1);
```

[动作]

- Var_D0＜0 的情况下，Var_D1 中将存储 (Var_D0)+(-100) 的值。
  例　Var_D0=-200 → Var_D1=-300
- Var_D0=0 的情况下，Var_D1 中将存储 0。
- 0＜Var_D0 的情况下，Var_D1 中将存储 (Var_D0)+100 的值。
  例　Var_D0=700 → Var_D1=800

(2) 以下为 X1 变为 ON 时，以 Var_D0 中设置的数据进行 -10000 ～ 10000 的区域控制后，将结果存储到 Var_D10 中的程序。

[结构体梯形图]



[ST]

```
DZONEP(X1,-10000,10000,Var_D0,Var_D10);
```

[动作]

- Var_D0＜0 的情况下，Var_D10 中将存储 Var_D0+(-10000) 的值。
  例　Var_D0=-12345 → Var_D10=-22345
- Var_D0=0 的情况下，Var_D10 中将存储 0。
- 0＜Var_D0 的情况下，Var_D10 中将存储 Var_D0+10000 的值。
  例　Var_D0=50000 → Var_D10=60000

7.13.4 标度（点坐标数据）

SCL
DSCL

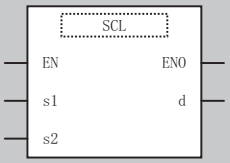
Basic　High performance　Ver. Universal　L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

SCL(P)
DSCL(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:=  SCL  (EN, s1, s2, d);
```

[　　] 中放入下述指令。

SCL	SCLP
DSCL	DSCLP

输入自变量，	EN:	执行条件	：位
	s1:	进行标度的输入值或存储输入值的软元件的起始编号	：ANY16/32
	s2:	存储标度用转换数据的软元件的起始编号	：ANY16/32
输出自变量，	ENO:	执行结果	：位
	d:	存储通过标度进行了控制的输出值的软元件的起始编号	：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K, H	
ⓢ1	－	○	○	○				○	－
ⓢ2	－	－	○	－				－	－
ⓓ	－	○	○	○				－	－

功 能

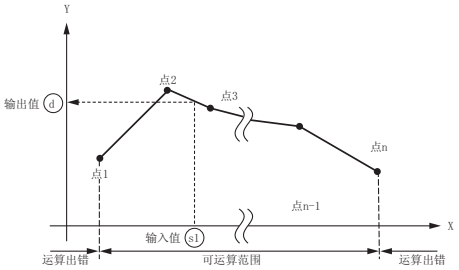
SCL(P)

(1) 对 ⓢ2 中指定的标度用转换数据（16 位数据单位）通过 ⓢ1 中指定的输入值进行标度后，将运算结果存储到 ⓓ 中指定的软元件编号中。
标度转换是根据 ⓢ2 中指定的软元件以后存储的标度用转换数据进行。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		ⓢ2
点1	X坐标	ⓢ2+1
	Y坐标	ⓢ2+2
点2	X坐标	ⓢ2+3
	Y坐标	ⓢ2+4
⋮		
点n	X坐标	ⓢ2+2n-1
	Y坐标	ⓢ2+2n

※n表示(s2)中指定的坐标点数。



(2) 运算结果不是整数值的情况下，将小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应以升序进行设置。

(4) ⓢ1 应在标度用转换数据范围内（ⓢ2 的软元件值）进行设置。

(5) 多个点指定了相同的 X 坐标的情况下，将对点 No. 最大的点的 Y 坐标的值进行输出。
标度用转换数据的坐标点数应在 1 ～ 32767 的范围内进行设置。

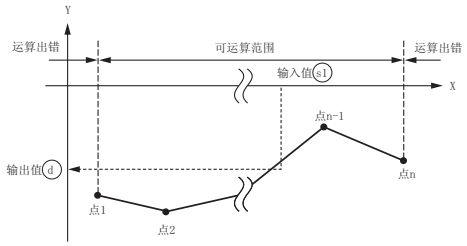
DSCL(P)

(1) 对 ⓢ2 中指定的标度用转换数据（32 位数据单位），通过 ⓢ1 中指定的输入值进行标度后，将运算结果存储到 ⓓ 中指定的软元件编号中。
标度转换是根据 ⓢ2 中指定的软元件以后存储的标度用转换数据进行。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		ⓢ2+1 ⓢ2
点1	X坐标	ⓢ2+3, ⓢ2+2
	Y坐标	ⓢ2+5, ⓢ2+4
点2	X坐标	ⓢ2+7, ⓢ2+6
	Y坐标	ⓢ2+9, ⓢ2+8
⋮		
点n	X坐标	ⓢ2+4n-1, ⓢ2+4n-2
	Y坐标	ⓢ2+4n+1, ⓢ2+4n

※n表示(s2)中指定的坐标点数。



(2) 运算结果不是整数值的情况下，将小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应以升序进行设置。

(4) ⓢ1 应在标度用转换数据范围内（ⓢ2， ⓢ2 +1 的软元件值）进行设置。

(5) 多个点指定了相同的 X 坐标的情况下将对点 No. 最大的点的 Y 坐标的值进行输出。

(6) 标度用转换数据的坐标点数应在 1 ～ 32767 的范围内进行设置。

要 点

(1) 查找方法根据 SM750 的 ON/OFF 状态而有所不同。

SM750	查找方法	查找次数范围
OFF	逐次查找	1 ≦次数≦ 32767
ON	二分查找	1 ≦次数≦ 15

(2) 标度用转换数据以升序排序的情况下，根据 SM750 的状态查找方法有所不同，因此处理速度也不相同。

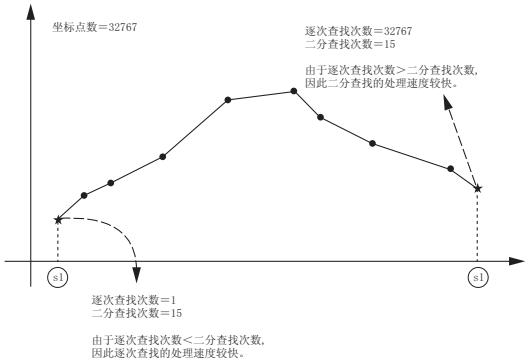
处理速度取决于查找次数，查找次数越少则处理速度越快。

(a) 逐次查找的处理速度较快的情况

岁在坐标点数最大，(s1) 位于坐标点 1 ～ 15 之间的情况下，由于逐次查找次数≦ 15，因此逐次查找的处理速度较快。

(b) 二分查找的处理速度较快的情况

由于最大查找次数为 15 次，因此在(s1) 位于坐标点 16 以后的情况下，二分查找次数≦逐次查找次数，因此二分查找的处理速度较快。



出 错

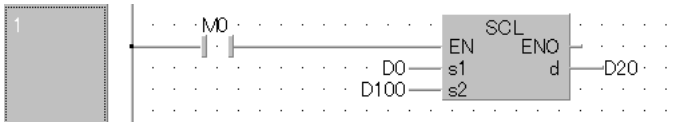
(1) 在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- (s1) 前面的点的标度用转换数据的 X 坐标数据未以升序设置时。(但是，SM750 为 ON 时，不会检测出本出错。) (出错代码： 4100)
- (s1) 中指定的输入值超出了所设置的标度用转换数据的范围时。 (出错代码： 4100)
- (s2) 的软元件算起的坐标点数超出了 1 ～ 32767 的范围时。 (出错代码： 4100)
- (s2) 的软元件算起的坐标点数超出了指定软元件范围时。 (出错代码： 4101)

程序示例

(1) 以下为 M0 变为 ON 时，用 D0 中指定的输入值对 D100 以后设置的标度用转换数据进行标度后，输出到 D20 中的程序。

[结构体梯形图]



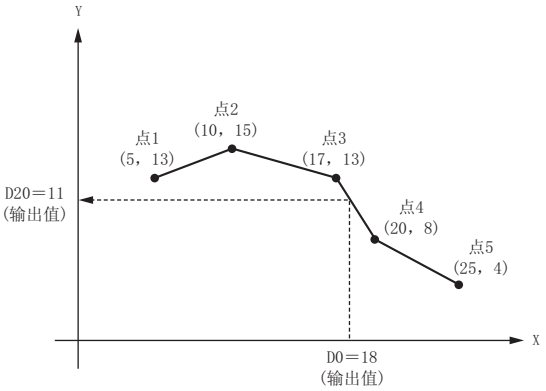
[ST]

```
SCL(M0,D0,D100,D20);
```

[动作]

标度用转换数据的构成

设置项目		软元件	设置内容
坐标点数		D100	K5
点1	X坐标	D101	K5
	Y坐标	D102	K13
点2	X坐标	D103	K10
	Y坐标	D104	K15
点3	X坐标	D105	K17
	Y坐标	D106	K13
点4	X坐标	D107	K20
	Y坐标	D108	K8
点5	X坐标	D109	K25
	Y坐标	D110	K22



7.13.5 标度（X/Y 坐标数据）

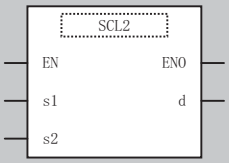
SCL2
DSCL2

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

SCL2(P)
DSCL2(P)

P：执行条件　　：

结构体梯形图



ST

```
ENO:= SCL2 (EN, s1, s2, d);
```

[ ] 中放入下述指令。

SCL2	SCL2P
DSCL2	DSCL2P

输入自变量，	EN:	执行条件	：位
	s1:	进行标度的输入值或存储输入值的软元件的起始编号	：ANY16/32
	s2:	存储标度用转换数据的软元件的起始编号	：ANY16/32
输出自变量，	ENO:	执行结果	：位
	d:	存储通过标度进行了控制的输出值的软元件的起始编号	：ANY16/32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K, H	
ⓢ1	－	○	○	○				○	－
ⓢ2	－	－	○	－				－	－
ⓓ	－	○	○	○				－	－

功 能

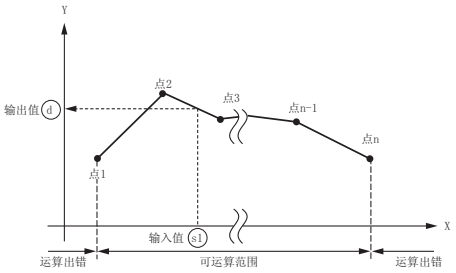
SCL2(P)

(1) 将 (s2) 中指定的标度用转换数据（16 位数据单位）通过 (s1) 中指定的输入值进行标度后，将运算结果存储到 (d) 中指定的软元件编号中。
标度转换是根据 (s2) 中指定的软元件以后存储的标度用转换数据进行。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(s2)
X坐标	点1	(s2)+1
	点2	(s2)+2
	⋮	⋮
	点n	(s2)+n
Y坐标	点1	(s2)+n+1
	点2	(s2)+n+2
	⋮	⋮
	点	(s2)+2n

※n表示(s2)中指定的坐标点数。



(2) 运算结果不是整数值的情况下，将小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应以升序进行设置。

(4) (s1) 应在标度用转换数据范围内（(s2) 的软元件值）进行设置。

(5) 多个点指定了相同的 X 坐标的情况下将对点 No. 最大的点的 Y 坐标的值进行输出。

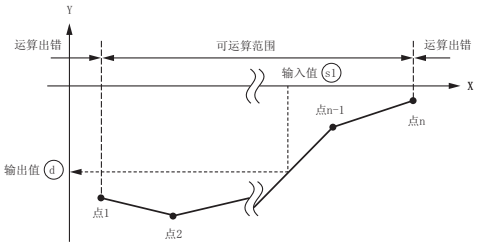
DSCL2(P)

(1) 对 (s2) 中指定的标度用转换数据（32 位数据单位），通过 (s1) 中指定的输入值进行标度后，将运算结果存储到 (d) 中指定的软元件编号中。
标度转换是根据 (s2) 中指定的软元件以后存储的标度用转换数据进行。

标度用转换数据的构成

设置项目		软元件分配
坐标点数		(s2)+1 , (s2)
X坐标	点1	(s2)+3 , (s2)+2
	点2	(s2)+5 , (s2)+4
	⋮	⋮
	点n	(s2)+2n+1 , (s2)+2n
Y坐标	点1	(s2)+2n+3 , (s2)+2n+2
	点2	(s2)+2n+5 , (s2)+2n+4
	⋮	⋮
	点n	(s2)+4n+1 , (s2)+4n

※n表示(s2)中指定的坐标点数。



(2) 运算结果不是整数值的情况下，将小数点以下第 1 位进行四舍五入。

(3) 标度用转换数据的 X 坐标数据应以升序进行设置。

(4) (s1) 应在标度用转换数据范围内（(s2)，(s2)+1 的软元件值）进行设置。

(5) 多个点指定了相同的 X 坐标的情况下将对点 No. 最大的点的 Y 坐标的值进行输出。

(6) 标度用转换数据的坐标点数应在 1 ～ 32767 的范围内进行设置。

要 点

在标度用转换数据以升序排序的情况下，查找方法根据 SM750 的状态而有所不同，因此处理速度也不相同。
详细内容请参阅 7.13.4 项。

出 错

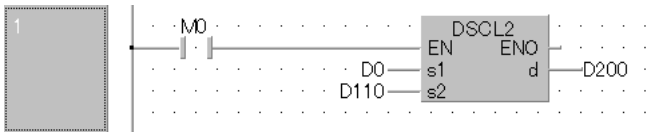
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- X 坐标数据未以升序设置时。(出错代码 : 4100)
- ⑤1 中指定的输入值超出了所设置的标度用转换数据的范围时。 (出错代码 : 4100)
- ⑤2 中指定的坐标点数超出了 1 ～ 32767 的范围时。 (出错代码 : 4100)
- ⑤2 的软元件算起的坐标点数超出了指定软元件范围时。 (出错代码 : 4101)

程序示例

以下为 M0 变为 ON 时，用 D0 中指定的输入值对 D110 以后设置的标度用转换数据进行标度后，输出到 D200 中的程序。

[结构体梯形图]



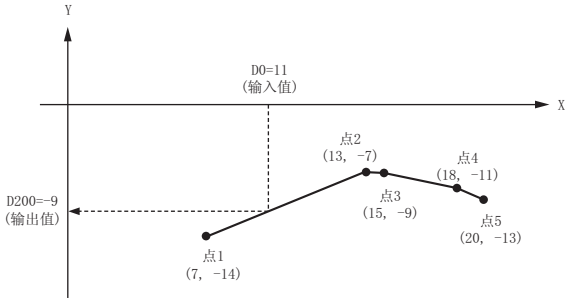
[ST]

```
DSCL2(M0,D0,D110,D200);
```

[动作]

标度用转换数据的构成

设置项目		软元件	设置内容
坐标点数		D110	K5
X坐标	点1	D111	K7
	点2	D112	K13
	点3	D113	K15
	点4	D114	K18
	点5	D115	K20
Y坐标	点1	D116	K-14
	点2	D117	K-7
	点3	D118	K-15
	点4	D119	K-11
	点5	D120	K-18



7.14 文件寄存器切换指令

7.14.1 文件寄存器的块号切换

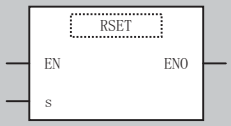
RSET

Basic | High performance | Universal | L CPU

RSET(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= RSET (EN, s);

□ 中放入下述指令。

RSET　　RSETP

输入自变量，EN:　执行条件　：位
　　s:　切换的块号数据或存储块号数据的软元件编号　：ANY16
输出自变量，ENO:　执行结果　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
Ⓢ	○								-

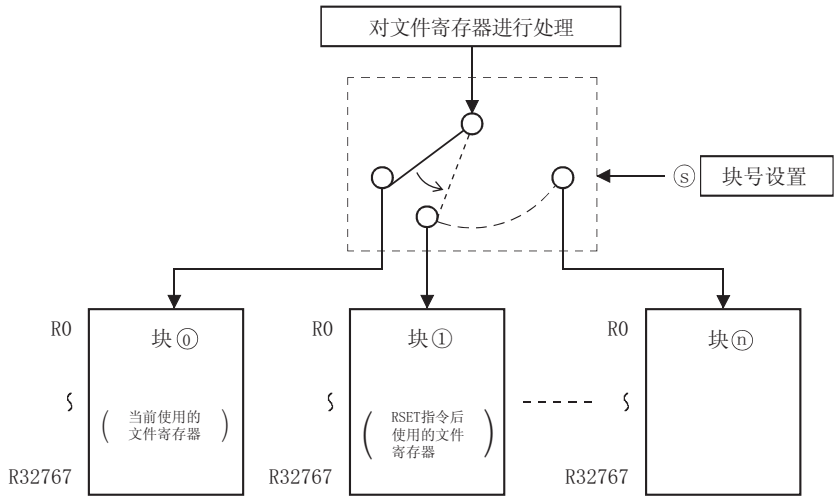
功 能

将程序中使用的文件寄存器的块号变更为Ⓢ中指定的软元件中存储的块号。

块号变更后，对于顺控程序中使用的所有的文件寄存器，按变更后的块号的文件寄存器进行处理。

例

从块号 0 切换为块号 1 时



要 点

将文件寄存器 (R) 指定为刷新软元件，通过 RSET(P) 对文件寄存器 (R) 的块号进行切换时应加以注意。

关于文件寄存器的限制事项，请参阅 3.5 节。

出 错

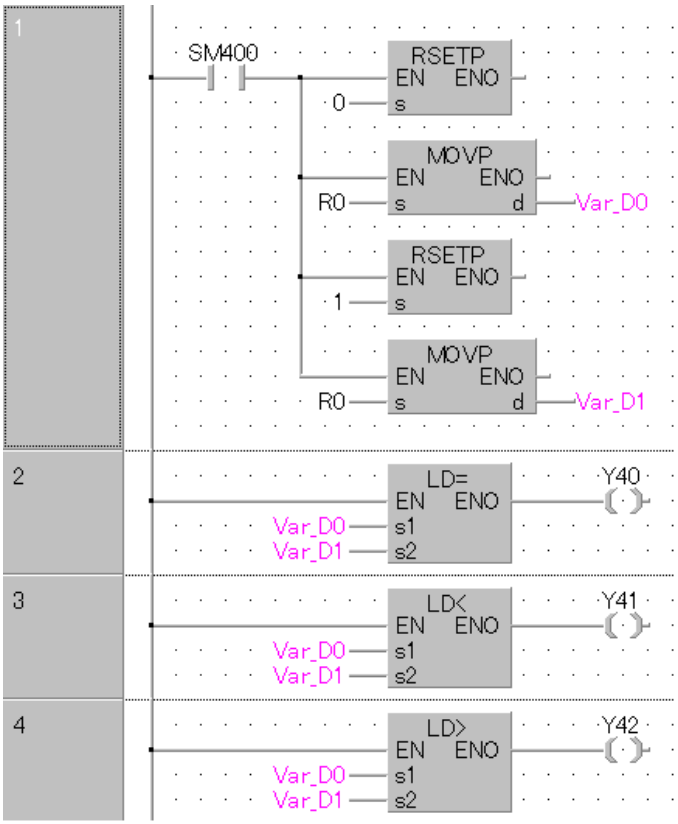
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ 中指定的块号不存在时。 (出错代码： 4100)
- 文件寄存器不存在时。 (出错代码： 4101)

程序示例

以下为将块号 0 与块号 1 的 R0 进行比较的程序。

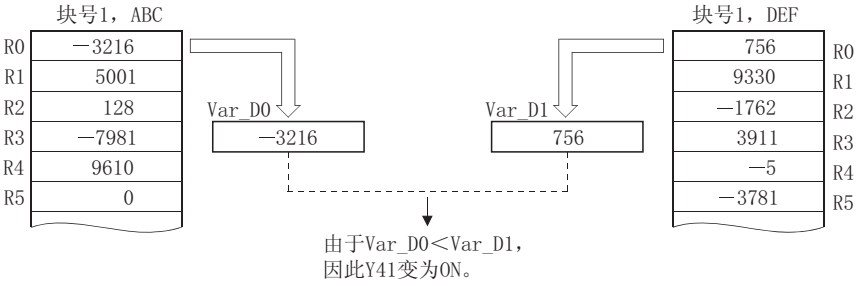
[结构体梯形图]



[ST]

```
IF SM400 THEN
    RSETP(TRUE,0);
    MOVP(TRUE,R0,Var_D0);
    RSETP(TRUE,1);
    MOVP(TRUE,R0,Var_D1);
END_IF;
OUT(Var_D0=Var_D1,Y40);
OUT(Var_D0<Var_D1,Y41);
OUT(Var_D0>Var_D1,Y42);
```

[动作]



7.14.2 文件寄存器用文件的设置

QDRSET

Basic　High performance　Universal　L CPU

QDRSET(P)

P: 执行条件　　: ↰

结构体梯形图

ST

ENO:= QDRSET (EN, s);

□ 中放入下述指令。

QDRSET　　QDRSETP

输入自变量， EN:　执行条件　：位
　　s:　设置的文件寄存器的驱动器 No. 文件名的字符串数据　：字符串
输出自变量， ENO:　执行结果　：位

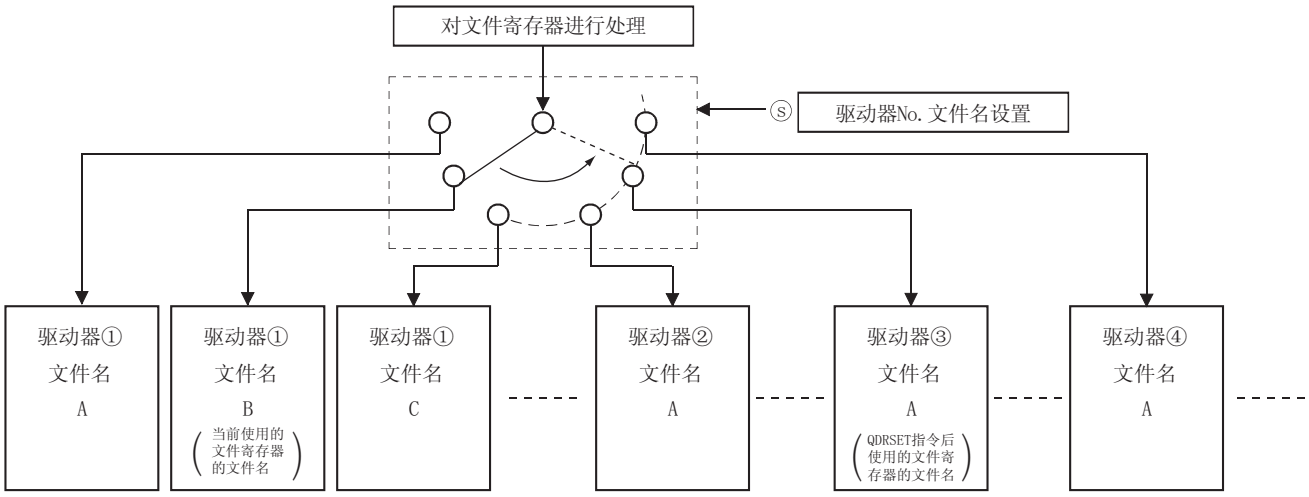
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	-	○		-				○	-

功 能

(1) 将程序中使用的文件寄存器的文件名变更为Ⓢ中指定的软元件中存储的文件名。
进行文件名变更后，对于顺控程序中使用的所有的文件寄存器，按变更后的文件名的块号 0 的文件寄存器进行处理。
块号的切换是通过 RSET(P) 指令进行。

例

将驱动器 No.1 的文件名 B 切换为驱动器 No.3 的文件名 A 时



(2) 驱动器 No. 的可指定范围为 1 ～ 4。
驱动器 No. 中不能指定驱动器 0(程序存储器 / 内置存储器)。

(3) 文件名中不需要指定扩展名 (.QDR)。

(4) 通过在文件名中指定 NULL 字符 (00H)，可以对文件名的设置进行解除。

(5) 即使在参数中已指定了驱动器 No.、文件名的情况下，本指令中指定的文件名也将优先。

要 点

1. 即使通过 QDRSET(P) 指令对文件名进行了变更，通过 CPU 模块的 STOP → RUN 操作，仍将恢复为参数中设置的文件名。
在希望进行了 CPU 模块的 STOP → RUN 后，通过 QDRSET(P) 指令变更的文件名仍然保持的情况下，在执行 QDRSET(P) 指令时，应使用 STOP → RUN 时 1 个扫描 ON 的特殊继电器 SM402。
2. 将文件寄存器指定为刷新软元件的情况下，不要通过 QDRSET(P) 指令对文件寄存器的文件名进行变更。
关于文件寄存器的限制事项，请参阅 3.5 节。

出 错

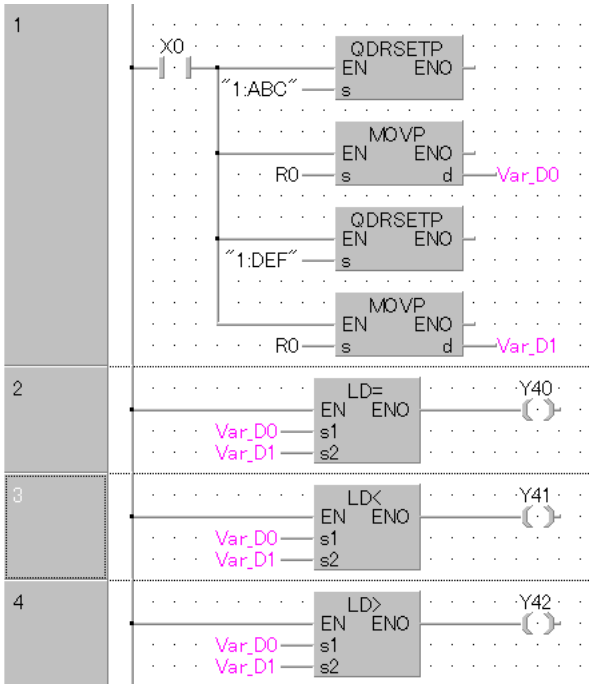
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的驱动器 No. 的文件名不存在时。 （出错代码： 2410）

程序示例

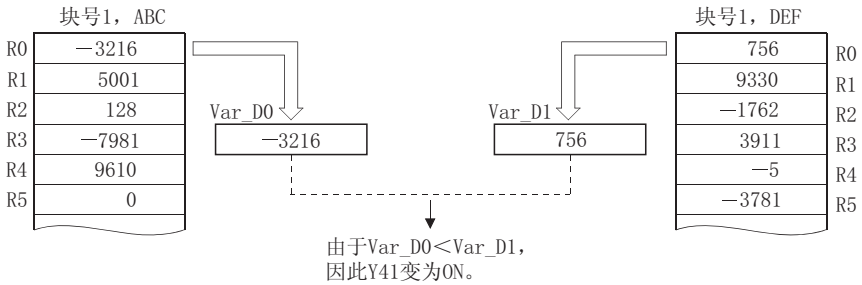
以下为将驱动器 No.1 的 ABC 与驱动器 No.1 的 DEF 的 R0 进行比较的程序。

[结构体梯形图]




```
[ST]
IF X0 THEN
     QDRSETP(TRUE,"1:ABC");
     MOVP(TRUE,R0,Var_D0);
     QDRSETP(TRUE,"1:DEF");
     MOVP(TRUE,R0,Var_D1);
END_IF;
OUT(Var_D0=Var_D1),Y40);
OUT(Var_D0<Var_D1),Y41);
OUT(Var_D0>Var_D1),Y42);
```

[动作]



7.14.3 注释用文件的设置

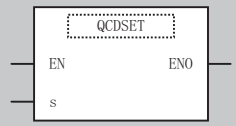
QCDSET

Basic | High performance | Universal | L CPU

QCDSET(P)

P: 执行条件 : ↰

结构体梯形图



ST

```
ENO:= QCDSET (EN, s);
```

[ ] 中放入下述指令。

QCDSET QCDSETP

输入自变量， EN: 执行条件 : 位
s: 设置的注释文件的驱动器No.文件名的字符串数据 : 字符串
输出自变量， ENO: 执行结果 : 位

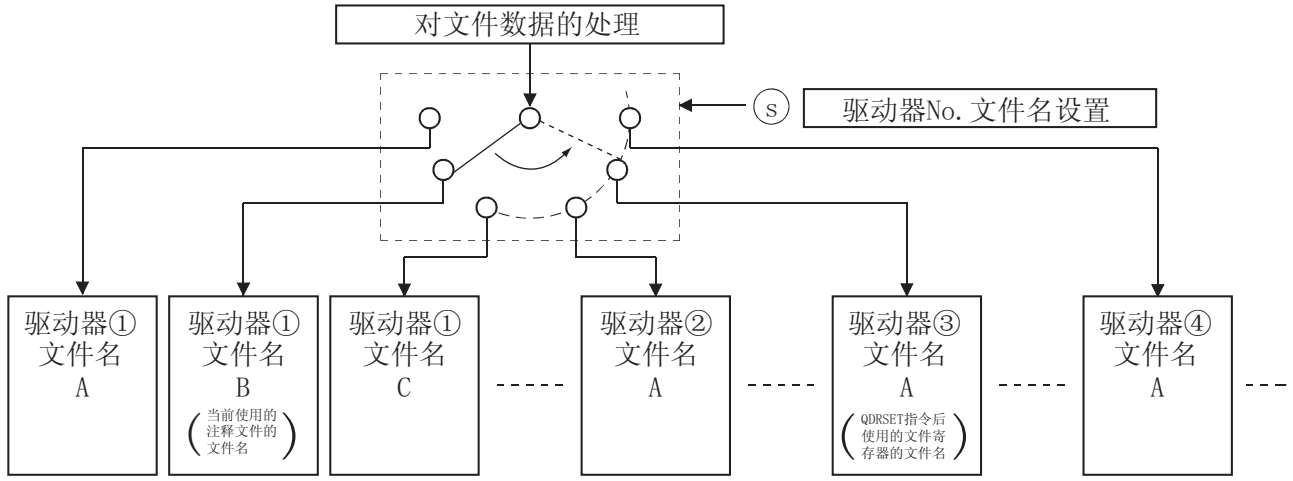
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它
	位	字		位	字				
Ⓢ	-	○		-				○	-

功 能

(1) 将程序中使用的注释文件的文件名变更为Ⓢ中指定的软元件中存储的文件名。
进行文件名变更后，对于顺控程序中使用的注释数据，将按变更后的文件名的注释数据进行处理。

例

从驱动器 No.1 的文件 B 切换为驱动器 No.3 的文件 A 时



(2) 驱动器 No. 的可指定范围为 1 ～ 4。
驱动器 No. 中不能指定驱动器 0(程序存储器 / 内置存储器)。
但是，根据 CPU 模块可指定的驱动器有所不同。
应通过所使用的 CPU 模块的手册对可指定的驱动器进行确认。

(3) 文件名不需要指定扩展名 (.QCD)。

(4) 通过在文件名中指定 NULL 字符 (00H)，可以对文件名的设置进行解除。

(5) 即使在参数中已指定了驱动器 No.、文件名的情况下，本指令中指定的文件名也将优先。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的驱动器 No. 的文件名不存在时。 (出错代码 : 2410)

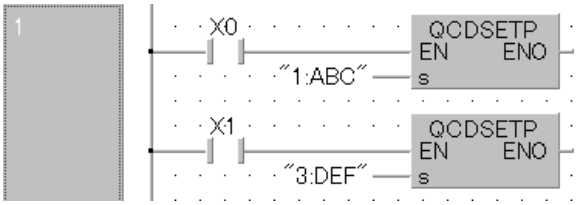
注意事项

在使用通用型 QCPU、LCPU 的情况下，在 SM721(文件访问中) 为 ON 的状态下，即使将本指令的执行指令置为 ON，也不执行本指令。应设置为在 SM721 为 OFF 时执行。

程序示例

以下为 X0 变为 ON 时，将注释的对象文件切换为驱动器 1 的 ABC.QCD，X1 变为 ON 时切换为驱动器 3 的 DEF.QCD 的程序。

[结构体梯形图]



[ST]

```
QCDSETP(X0,"1:ABC");
QCDSETP(X1,"3:DEF");
```

要 点

即使通过 QCDSET(P) 指令对文件名进行了变更，通过 CPU 模块的 STOP → RUN 操作，仍将恢复为参数中设置的文件名。

在希望进行了 CPU 模块的 STOP → RUN 后，通过 QCDSET(P) 指令变更的文件名仍然保持的情况下，在执行 QCDSET(P) 指令时，应使用 STOP → RUN 时 1 个扫描 ON 的特殊继电器 SM402。

7.15 时钟用指令

7.15.1 时钟数据的读取

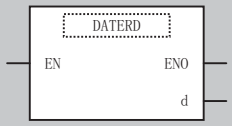
DATERD

Basic | High performance | Universal | L CPU

DATERD(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= DATERD (EN, d);

中放入下述指令。

DATERD DATERDP

输入自变量, EN: 执行条件 : 位
输出自变量, ENO: 执行结果 : 位
d: 读取的时钟数据 : ANY16 的数组 (0..6)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓓ	-	○		-					

功 能

(1) 通过 CPU 模块的时钟因子对年、月、日、分、秒、星期进行读取后，以 BIN 值存储到ⓓ中指定的软元件以后。

时钟因子			
	ⓓ[0]	年(公历)	(1980～2079)
	ⓓ[1]	月	(1～12)
	ⓓ[2]	日	(1～31)
	ⓓ[3]	时(24小时制)	(0～23)
	ⓓ[4]	分	(0～59)
	ⓓ[5]	秒	(0～59)
	ⓓ[6]	星期	(0～6)

(2) ⓓ的年以公历 4 位进行存储。

(3) ⓓ+6 的星期的周日～周六以 0 ～ 6 进行存储。

星期	日	一	二	三	四	五	六
存储数据	0	1	2	3	4	5	6

(4) 闰年将自动修正。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

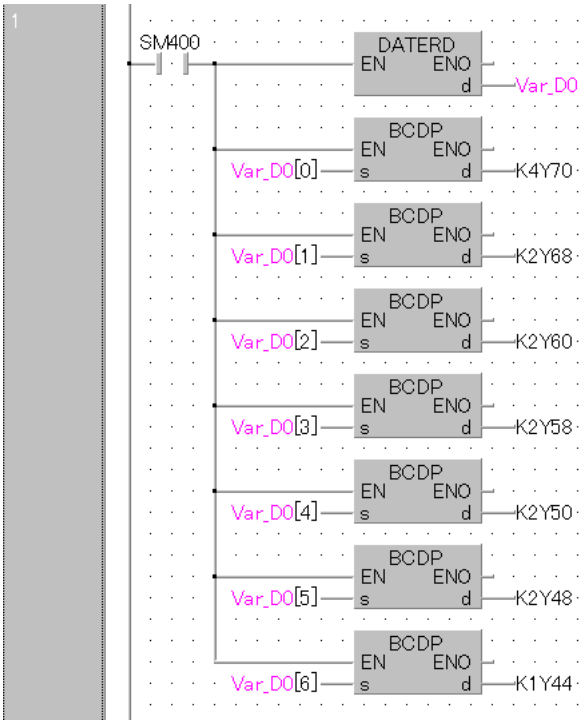
- ⓓ中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码 : 4101)

程序示例

以下为对下述时钟数据以 BCD 值进行输出的程序。

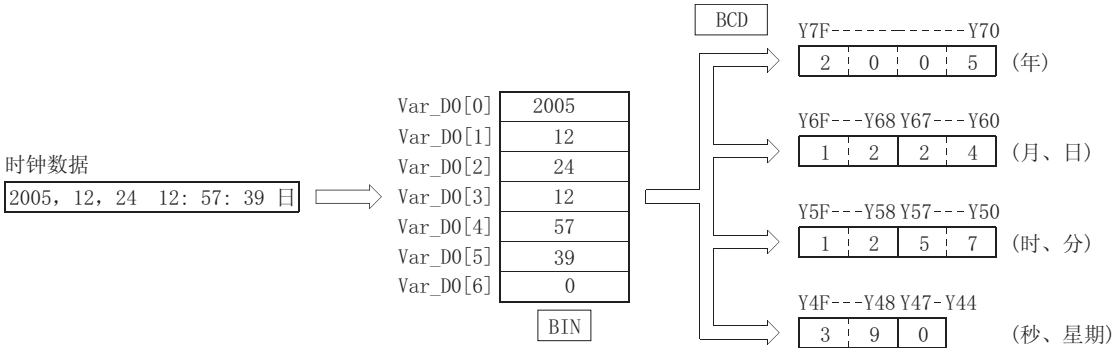
年.....Y70 ～ Y7F
月.....Y68 ～ Y6F
日.....Y60 ～ Y67
时.....Y58 ～ Y5F
分.....Y50 ～ Y57
秒.....Y48 ～ Y4F
星期.....Y44 ～ Y47

[结构体梯形图]



```
[ST]
IF SM400 THEN
     DATERD(TRUE,Var_D0);
     BCDP(TRUE,Var_D0[0],K4Y70);
     BCDP(TRUE,Var_D0[1],K2Y68);
     BCDP(TRUE,Var_D0[2],K2Y60);
     BCDP(TRUE,Var_D0[3],K2Y58);
     BCDP(TRUE,Var_D0[4],K2Y50);
     BCDP(TRUE,Var_D0[5],K2Y48);
     BCDP(TRUE,Var_D0[6],K1Y44);
END_IF;
```

[动作]



7.15.2 时钟数据的写入

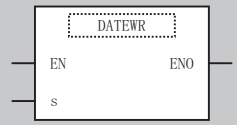
DATEWR

Basic | High performance | Universal | L CPU

DATEWR(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= DATEWR (EN, s);

中放入下述指令。

DATEWR DATEWRP

输入自变量， EN: 执行条件 : 位
s: 写入到时钟因子的时钟数据 : ANY16 的数组 (0..6)
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	-	○		-					

功 能

(1) 将ⓢ中指定的软元件编号以后存储的时钟数据，写入到 CPU 模块的时钟因子中。

ⓢ[0]	年
ⓢ[1]	月
ⓢ[2]	日
ⓢ[3]	时
ⓢ[4]	分
ⓢ[5]	秒
ⓢ[6]	星期

⇒ 时钟因子

(2) 各项目的设置是以 BIN 值进行设置。

(3) ⓢ的年是以公历 4 位在 1980 ～ 2079 的范围内进行设置。
(如果以年号进行设置，将不能进行闰年的自动修正。)

(4) ⓢ+1 的月是以 1 ～ 12(1 月～ 12 月) 进行设置。

(5) Ⓢ+2 的日是以 1 ～ 31(1 日～ 31 日) 进行设置。

(6) Ⓢ+3 的时是以 0 ～ 23(0 时～ 23 时) 进行设置。
(以 24 小时制进行设置。)

(7) Ⓢ+4 的分是以 0 ～ 59(0 分～ 59 分) 进行设置。

(8) Ⓢ+5 的秒是以 0 ～ 59(0 秒～ 59 秒) 进行设置。

(9) Ⓢ+6 的星期是以周日～周六 (0 ～ 6) 进行设置。

星期	日	一	二	三	四	五	六
存储数据	0	1	2	3	4	5	6

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

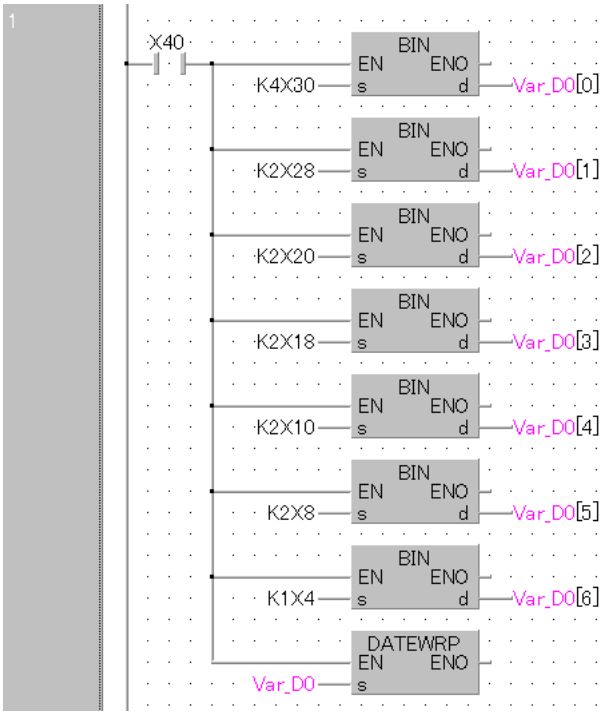
- 各项目中设置了超出设置范围的数据时。 (出错代码 : 4100)
- Ⓢ中指定的软元件超出了相应软元件的范围时。
(通用型 QCPU、LCPU 时) (出错代码 : 4101)

程序示例

以下为 X40 变为 ON 时，将以 BCD 值输入的下述时钟数据写入到时钟因子中的程序。

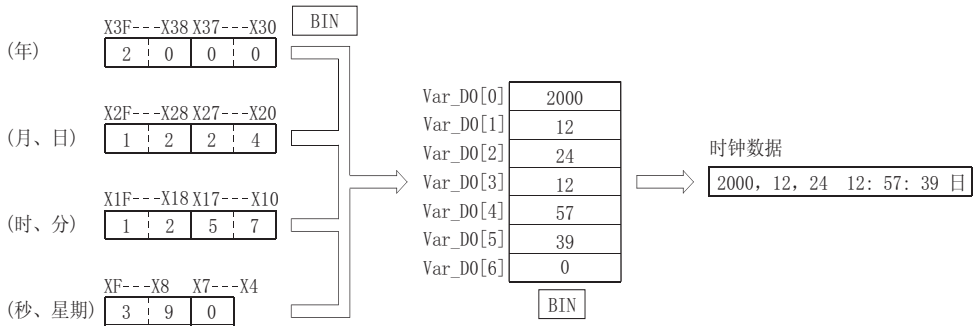
年 . . . . . X30 ～ X3F　　时 . . . . . X18 ～ X1F
月 . . . . . X28 ～ X2F　　分 . . . . . X10 ～ X17
日 . . . . . X20 ～ X27　　秒 . . . . . X8 ～ X7
星期 . . . X4 ～ X7

[结构体梯形图]



```
[ST]
IF X40 THEN
     BIN(TRUE,K4X30,Var_D0[0]);
     BIN(TRUE,K2X28,Var_D0[1]);
     BIN(TRUE,K2X20,Var_D0[2]);
     BIN(TRUE,K2X18,Var_D0[3]);
     BIN(TRUE,K2X10,Var_D0[4]);
     BIN(TRUE,K2X8,Var_D0[5]);
     BIN(TRUE,K1X4,Var_D0[6]);
     DATEWRP(TRUE,Var_D0);
END_IF;
```

[动作]



7.15.3 时钟数据的加法

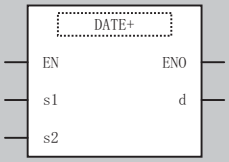
DATE+

Basic | High performance | Universal | L CPU

DATE+(P)

P: 执行条件 : ↰

结构体梯形图



ST

不支持。

□ 中放入下述指令。

DATE+ DATE+P

输入自变量， EN: 执行条件 : 位
s1: 被进行加法运算的时间（时间）数据 : ANY16 的数组 (0..2)
s2: 加法时间数据 : ANY16 的数组 (0..2)
输出自变量， ENO: 执行结果 : 位
d: 加法结果时间（时间）数据 : ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ1	-	○		-					
ⓢ2	-	○		-					
ⓓ	-	○		-					

功 能

(1) 将ⓢ1中指定的时间数据与ⓢ2中指定的时间数据进行加法运算后，将加法结果存储到ⓓ中指定的软元件编号以后。

		数据范围			数据范围			数据范围
ⓢ1[0]	时	(0～23)	ⓢ2[0]	小时	(0～23)	ⓓ[0]	时	(0～23)
ⓢ1[1]	分	(0～59) +	ⓢ2[1]	分	(0～59) ⇨	ⓓ[1]	分	(0～59)
ⓢ1[2]	秒	(0～59)	ⓢ2[2]	秒	(0～59)	ⓓ[2]	秒	(0～59)

例如将 6 时 32 分 40 秒与 7 小时 48 分 10 秒进行加法运算时，其情况如下所示。

ⓢ1[0]	6时		ⓢ2[0]	7小时		ⓓ[0]	14时
ⓢ1[1]	32分	+	ⓢ2[1]	48分	⇨	ⓓ[1]	20分
ⓢ1[2]	40秒		ⓢ2[2]	10秒		ⓓ[2]	50秒

(2) 运算结果时间超过了 24 时的情况下，运算结果将变为减去 24 小时后的值。

例如将 14 时 20 分 30 秒与 20 小时 20 分 20 秒进行加法运算的情况下，其结果不是 34 时 40 分 50 秒，而是 10 时 40 分 50 秒。

ⓢ1 [0]	14时		ⓢ2 [0]	20小时	⇒	ⓓ [0]	10时
ⓢ1 [1]	20分	+	ⓢ2 [1]	20分		ⓓ [1]	40分
ⓢ1 [2]	30秒		ⓢ2 [2]	20秒		ⓓ [2]	50秒

备 注

关于时、分、秒中可设置的数据，请参阅 7.15.2 项。

出 错

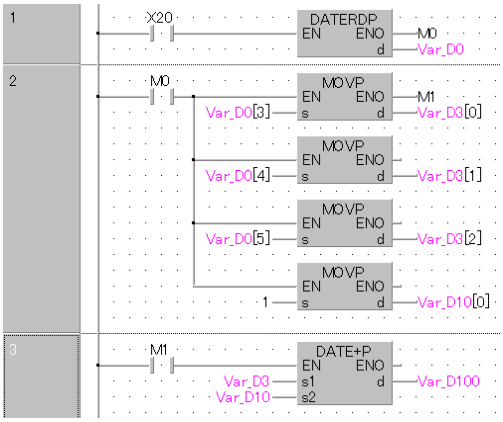
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1，ⓢ2 的数据超出了范围时。 （出错代码： 4100）
- ⓢ1，ⓢ2，ⓓ 中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，在从时钟因子中读取的时间数据中加上 1 小时后，将运算结果存储到 Var_D100 以后的程序。

[结构体梯形图]



[动作]

• 通过 DATERDP 指令读取时间数据

时钟因子	⇒				
		Var_D0[0]	95	年	
		Var_D0[1]	5	月	
		Var_D0[2]	15	日	
		Var_D0[3]	10	时	时间数据
		Var_D0[4]	23	分	
		Var_D0[5]	41	秒	
		Var_D0[6]	2	星期	

• 通过 DATE+ 指令进行加法运算

Var_D3[3]	10时		Var_D10[0]	1小时	⇒	Var_D100[0]	11时
Var_D3[4]	23分	+	Var_D10[1]	0分		Var_D100[1]	23分
Var_D3[5]	41秒		Var_D10[2]	0秒		Var_D100[2]	41秒

7.15.4 时钟数据的减法

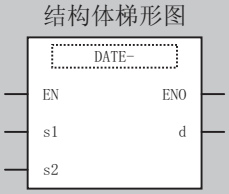
DATE-

Basic | High performance | Universal | L CPU

DATE-(P)

P: 执行条件 : ↰

结构体梯形图



ST

不支持。

□ 中放入下述指令。

DATE- DATE-P

输入自变量， EN: 执行条件 : 位
s1: 被进行减法运算的时间（时间）数据 : ANY16 的数组 (0..2)
s2: 减法时间数据 : ANY16 的数组 (0..2)
输出自变量， ENO: 执行结果 : 位
d: 减法结果时间（时间）数据 : ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ1	-	○		-					
ⓢ2	-	○		-					
ⓓ	-	○		-					

功 能

(1) 将ⓢ1中指定的时间数据与ⓢ2中指定的时间数据进行减法运算，将减法结果存储到ⓓ中指定的软元件编号以后。

		数据范围			数据范围			数据范围
ⓢ1[0]	时	(0～23)	ⓢ2[0]	小时	(0～23)	ⓓ[0]	时	(0～23)
ⓢ1[1]	分	(0～59)	ⓢ2[1]	分	(0～59)	ⓓ[1]	分	(0～59)
ⓢ1[2]	秒	(0～59)	ⓢ2[2]	秒	(0～59)	ⓓ[2]	秒	(0～59)

例如从 10 时 40 分 20 秒中减去 3 小时 50 分 10 秒时的情况如下所示。

ⓢ1[0]	10时	ⓢ2[0]	3小时	ⓓ[0]	6时
ⓢ1[1]	40分	ⓢ2[1]	50分	ⓓ[1]	50分
ⓢ1[2]	20秒	ⓢ2[2]	10秒	ⓓ[2]	10秒

(2) 运算结果时间为负数的情况下，运算结果将变为该负数数据加上 24 以后的值。

例如从 4 时 50 分 32 秒中减去 10 小时 42 分 12 秒的情况下，其结果不是 -6 时 8 分 20 秒，而是 18 时 8 分 20 秒。

ⓢ1[0]	4时		ⓢ2[0]	10小时		ⓓ[0]	18时
ⓢ1[1]	50分	−	ⓢ2[1]	42分	⇨	ⓓ[1]	8分
ⓢ1[2]	32秒		ⓢ2[2]	12秒		ⓓ[2]	20秒

备 注

关于时、分、秒中可设置的数据，请参阅 7.15.2 项。

出 错

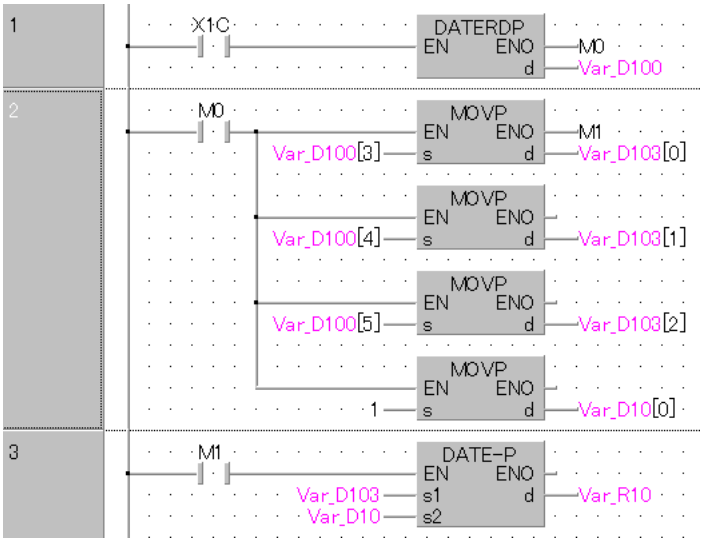
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1，ⓢ2 的数据超出了范围时。 （出错代码： 4100）
- ⓢ1，ⓢ2，ⓓ 中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

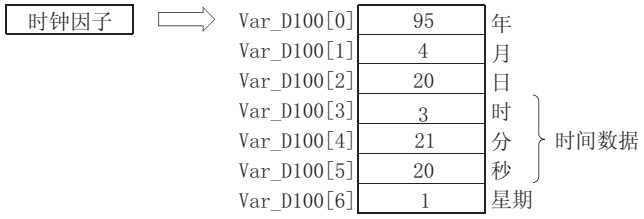
以下为 X1C 变为 ON 时，将从时钟因子中读取的时间数据与 Var_D10 以后存储的时间数据进行减法运算后，将运算结果存储到 Var_R10 以后的程序。

[结构体梯形图]

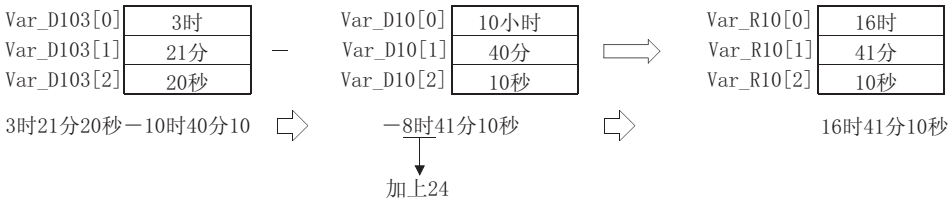


[动作]

• 通过 DATERDP 指令读取时间数据



• 通过 DATE- 指令进行减法运算（在 Var_D10 中指定了 10 小时 40 分 10 秒的情况下）



7.15.5 时间数据的转换（时、分、秒→秒）

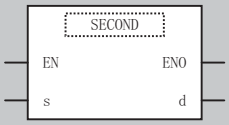
SECOND

Basic | High performance | Universal | L CPU

SECOND(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= SECOND (EN, s, d);

[ ] 中放入下述指令。

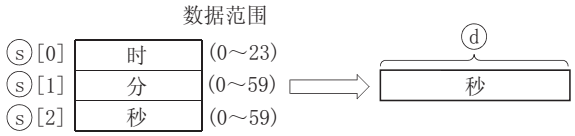
SECOND SECONDP

输入自变量， EN: 执行条件 : 位
s: 转换前的时钟数据 : ANY16 的数组 (0..2)
输出自变量， ENO: 执行结果 : 位
d: 转换后的时钟数据 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字				
ⓢ	–	○		–				–	
ⓓ	○	○		○				–	

功 能

将ⓢ中指定的软元件编号以后中存储的时间数据换算为秒后，将运算结果存储到ⓓ中指定的软元件中。



例如指定了 4 小时 29 分 31 秒的情况如下所示。

出 错

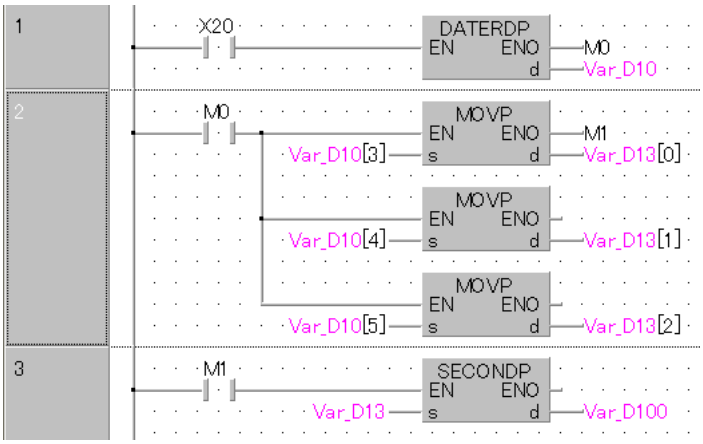
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- Ⓢ的数据超出了范围时。（出错代码： 4100）
- Ⓢ的文件名存储目标软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，将从时钟因子中读取的时间数据换算为秒后，存储到 Var_D100 中的程序。

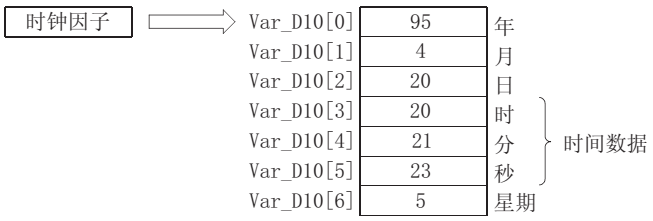
[结构体梯形图]



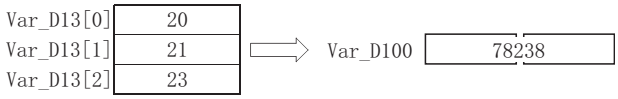

```
[ST]
IF X20 THEN
     DATERDP(TRUE,Var_D10);
     MOVP(TRUE,Var_D10[3],Var_D13[0]);
     MOVP(TRUE,Var_D10[4],Var_D13[1]);
     MOVP(TRUE,Var_D10[5],Var_D13[2]);
     SECONDP(TRUE,Var_D13,Var_D100);
END_IF;
```

[动作]

- 通过 DATERDP 指令读取时间数据



- 通过 SECONDP 指令换算为秒



7.15.6 时间数据的转换（秒→时、分、秒）

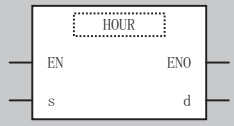
HOUR

Basic | High performance | Universal | L CPU

HOUR(P)

P: 执行条件 : ↑

结构体梯形图



ST

ENO:= HOUR (EN, s, d);

[ ] 中放入下述指令。

HOUR HOURP

输入自变量， EN: 执行条件 : 位
s: 转换前的时钟数据 : ANY32
输出自变量， ENO: 执行结果 : 位
d: 转换后的时钟数据 : ANY16 的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它
	位	字		位	字				
ⓢ	○	○		○				○	-
ⓓ	-	○		-				-	-

功 能

将ⓢ中指定的软元件编号中存储的秒数据换算为时、分、秒后，将换算结果存储到ⓓ中指定的软元件以后。

例如指定了 45325 秒的情况如下所示。

出 错

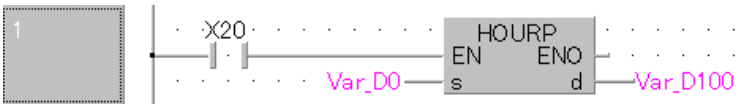
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ的数据超出了范围时。（出错代码： 4100）
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101）

程序示例

以下为 X20 变为 ON 时，将 Var_D0 中存储的秒数据换算为时、分、秒后，存储到 Var_D100 以后中的程序。

[结构体梯形图]

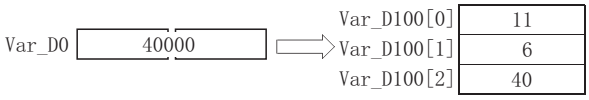


[ST]

```
HOURP(X20,Var_D0,Var_D100);
```

[动作]

• 通过 HOURP 指令转换为时、分、秒（在 Var_D0 中指定了 40000 秒时）



7.15.7 日期比较

□DT=, □DT<>, □DT<=, □DT<, □DT>=, □DT>

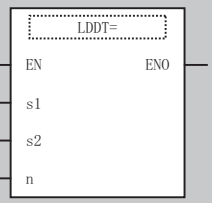
Basic × | High performance × | Ver. Universal | L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102”以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102”以后

LD	(DT=)
AND	(DT<>)
OR	(DT<=)
	(DT<)
	(DT>=)
	(DT>)

□DT=	:	=
□DT<>	:	≠
□DT<=	:	≤
□DT<	:	<
□DT>=	:	≥
□DT>	:	>

结构体梯形图



ST

ENO:= LDDT= (EN, s1, s2, n);

□ 中放入下述指令。

LDDT=	ANDDT=	ORDT=
LDDT<>	ANDDT<>	ORDT<>
LDDT<=	ANDDT<=	ORDT<=
LDDT<	ANDDT<	ORDT<
LDDT>=	ANDDT>=	ORDT>=
LDDT>	ANDDT>	ORDT>

输入自变量，	EN:	执行条件	: 位
	s1:	存储比较数据的软元件的起始编号	: ANY16
	s2:	存储被比较数据的软元件的起始编号	: ANY16
输出自变量，	ENO:	执行结果	: 位
	n:	表示比较对象的值或存储比较对象的数据数	: ANY16

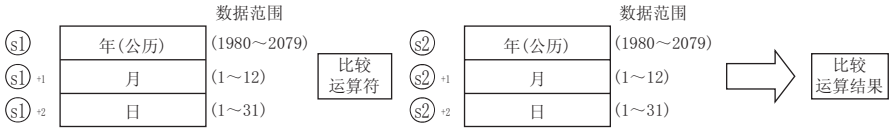
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K, H	
⑤1	–	○		–				–	–
⑤2	–	○		–				–	–
n	–	○		○				○	–

功 能

(1) 对 ⓢ1, ⓢ2 中指定的日期数据进行比较。或将 ⓢ1 中指定的日期数据与当前日期进行比较。通过 n 可以选择比较对象。

(a) 与任意的日期数据的比较

- 按照 n 的条件通过常开触点处理对 ⓢ1 中指定的日期数据与 ⓢ2 中指定的日期数据进行比较。



(b) 与当前的日期数据的比较

- 按照 n 的条件通过常开触点处理对 ⓢ1 中指定的日期数据与当前的日期数据进行比较。

要 点

与任意的日期数据进行比较时，或与当前的日期数据进行比较时，ⓢ1， ⓢ2 中之一处于下述情况下将可能发生运算出错（出错代码： 4101）或误动作。

- 变址修饰目标指定了超出软元件范围的范围时。
- 在未设置文件寄存器的状态下指定了文件寄存器时。

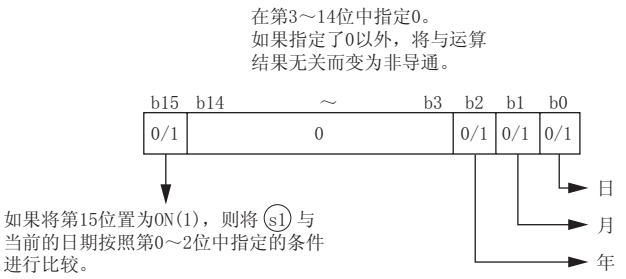
(2) 各项目的设置是以 BIN 值进行设置的。

(3) ⓢ1, ⓢ2 中的 “年”是以公历 4 位在 1980 ～ 2079 的范围内进行设置。

(4) ⓢ1 +1, ⓢ2 +1 中的 “月”是以 1 ～ 12(1 月～ 12 月）进行设置。

(5) ⓢ1 +2, ⓢ2 +2 中的 “日”是以 1 ～ 31(1 日～ 31 日）进行设置。

(6) 通过在 n 中指定下图的值，可以对比较对象进行详细指定。
n 的位构成如下所示。



(a) 比较对象日期（第 0 ～ 2 位）

- 0：不对比较对象的日期数据（年 / 月 / 日）进行比较。
- 1：对比较对象的日期数据（年 / 月 / 日）进行比较。

(b) 比较运算对象（第 15 位）

- 0：进行 ⓢ1 中指定的日期数据与 ⓢ2 中指定的日期数据的比较。
- 1：进行 ⓢ1 中指定的日期数据与当前的日期数据的比较。
  ⓢ2 中指定的日期数据将被忽略。

(c) 比较对象位的处理内容如下述所示。

与任意的日期数据比较时的 n 值	与当前的日期数据比较时的 n 值	比较对象日期	处理内容
0001H	8001H	日	仅对日（ⓢ1 +2）进行比较。
0002H	8002H	月	仅对月（ⓢ1 +1）进行比较。
0003H	8003H	月、日	对月（ⓢ1 +1），日（ⓢ1 +2）进行比较。
0004H	8004H	年	仅对年（ⓢ1）进行比较。
0005H	8005H	年、日	对年（ⓢ1），日（ⓢ1 +2）进行比较。
0006H	8006H	年、月	对年（ⓢ1），月（ⓢ1 +1）进行比较。
0007H	8007H	年、月、日	对年（ⓢ1），月（ⓢ1 +1），日（ⓢ1 +2） 全部进行比较。
0001H ～ 0007H, 8001H ～ 8007H 以外		无	不对年（ⓢ1），月（ⓢ1 +1），日（ⓢ1 +2） 进行全部比较。（变为非导通。）

(7) 比较对象软元件中存储的数据不能被识别为日期数据的情况下，在指令执行后将 SM709 置为 ON 后，变为非导通状态。即使不能被识别为日期数据的情况下，只要在设置范围内 SM709 将不变为 ON。

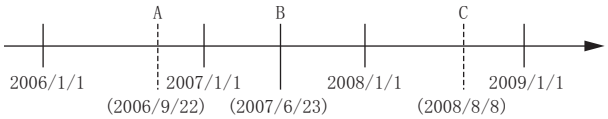
ⓢ1 ～ ⓢ1 +2 或 ⓢ2 ～ ⓢ2 +2 超出了指定软元件范围的情况下，将 SM709 置为 ON 后，变为非导通状态。

一旦将 SM709 置为 ON，在复位 / 电源 OFF 之前将保持为 ON 状态，因此应根据需要将其置为 OFF。

(8) 各指令的比较运算结果如下所示。

□内指令符号	条件	比较运算结果	□内指令符号	条件	比较运算结果
□DT=	ⓢ1 =ⓢ2	导通状态	□DT=	ⓢ1 ≠ⓢ2	非导通状态
□DT<>	ⓢ1 ≠ⓢ2		□DT<>	ⓢ1 =ⓢ2	
□DT>	ⓢ1 >ⓢ2		□DT>	ⓢ1 ≤ⓢ2	
□DT<=	ⓢ1 ≤ⓢ2		□DT<=	ⓢ1 >ⓢ2	
□DT<	ⓢ1 <ⓢ2		□DT<	ⓢ1 ≥ⓢ2	
□DT>=	ⓢ1 ≥ⓢ2		□DT>=	ⓢ1 <ⓢ2	

(a) 日期的比较示例如下所示。



上述所示的日期 A、B，C 的比较运算结果如下所示。
即使以相同条件进行了比较的情况下，根据选择的比较对象其比较运算结果将有所不同。

比较对象	比较条件		
	A<B	B<C	A<C
日	○	×	×
月	×	○	×
月、日	×	○	×
年	○	○	○
年、日	○	○	○
年、月	○	○	○
年、月、日	○	○	○
无	×	×	×

○：导通　×：非导通

(b) 即使比较的日期为实际中不存在的日期的情况下，只要是可设置的日期，将按照下述条件进行比较运算。

- 日期 A：2006/02/30（实际中不存在，但可进行日期设置。）
- 日期 B：2007/03/29
- 日期 C：2008/02/31（实际中不存在，但可进行日期设置。）

比较对象	比较条件		
	A<B	B<C	A<C
日	×	×	○
月	×	×	×
月、日	○	×	○
年	○	○	○
年、日	○	○	○
年、月	○	○	○
年、月、日	○	○	○
无	×	×	×

○：导通　×：非导通

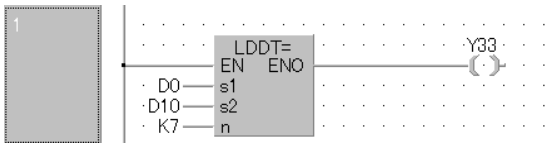
出 错

(1) 不存在 DT=、DT<>、DT>、DT<=、DT<、DT>= 指令相关的运算出错。

程序示例

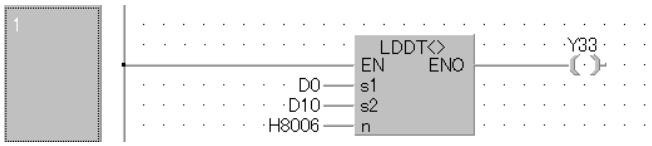
(1) 以下为将 D0 的数据与 D10 的数据（年、月、日）进行比较，D0 的数据与 D10 的数据一致时将 Y33 置为 ON 的程序。

[结构体梯形图]



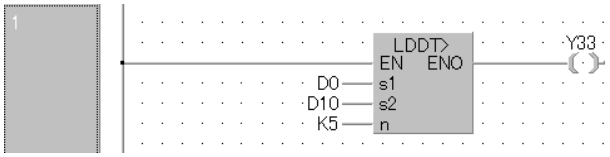
(2) 以下为 M0 变为 ON 时，将 D0 的数据与当前的日期数据（年、月）进行比较，D0 的数据与当前的日期数据不一致时将 Y33 置为 ON 的程序。

[结构体梯形图]



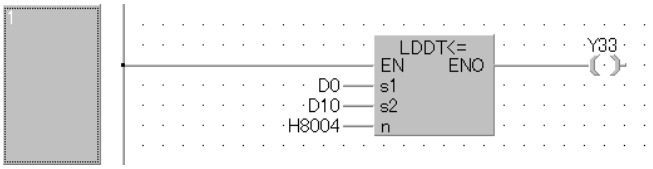
(3) 以下为 M0 变为 ON 时，将 D0 的数据与 D10 的数据（年、日）进行比较，D10 的数据低于 D0 时将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 D0 的数据与当前的日期数据（年）进行比较，当前的日期数据大于 D0 时将 Y33 置为 ON 的程序。

[结构体梯形图]



7.15.8 时间比较

□TM=, □TM<>, □TM<=, □TM<, □TM>=, □TM>

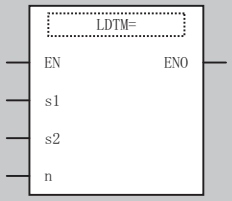
Basic ✕ High performance ✕ Ver. Universal L CPU

- Q00UJCPU、Q00UCPU、Q01UCPU
- QnU(D)(H)CPU：序列号的前 5 位数为 “10102” 以后
- QnUDE(H)CPU：序列号的前 5 位数为 “10102” 以后

LD	(TM=)
AND	(TM<>)
OR	(TM<=)
	(TM<)
	(TM>=)
	(TM>)

□TM=	:	=
□TM<>	:	≠
□TM<=	:	≤
□TM<	:	<
□TM>=	:	≥
□TM>	:	>

结构体梯形图



ST

不支持。

[ ] 中放入下述指令。

LDTM=	ANDTM=	ORTM=
LDTM<>	ANDTM<>	ORTM<>
LDTM<=	ANDTM<=	ORTM<=
LDTM<	ANDTM<	ORTM<
LDTM>=	ANDTM>=	ORTM>=
LDTM>	ANDTM>	ORTM>

输入自变量，	EN:	执行条件	: 位
	s1:	存储比较数据的软元件的起始编号	: ANY16
	s2:	存储比较数据的软元件的起始编号	: ANY16
输出自变量，	ENO:	执行结果	: 位
	n:	表示比较方法的值或存储比较方法的数据数	: ANY16

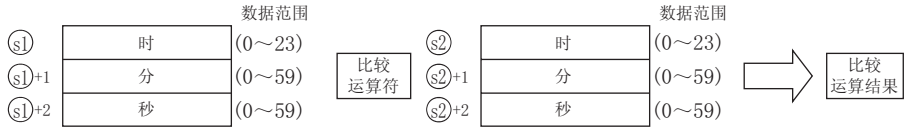
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它
	位	字		位	字			K, H	
ⓢ1	–	○		–				–	–
ⓢ2	–	○		–				–	–
n	–	○		○				○	–

功 能

(1) 对 ⓢ1, ⓢ2 中指定的时间数据进行比较。或对 ⓢ1 中指定的时间数据与当前时间进行比较。
通过 n 可以选择比较对象。

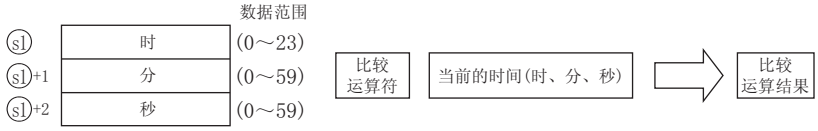
(a) 与任意的时间数据的比较

- 按照 n 的条件通过常开触点处理将 ⓢ1 中指定的时间数据与 ⓢ2 中指定的时间数据进行比较。



(b) 与当前的时间数据的比较

- 按照 n 的条件通过常开触点处理将 ⓢ1 中指定的时间数据与当前的时间数据进行比较。
- ⓢ2 中指定的时间数据将被作为虚拟数据处理而忽略。



要 点

与任意的时间数据进行比较时，或与当前的时间数据进行比较时，ⓢ1, ⓢ2 中之一处于下述情况下将可能发生运算出错（出错代码：4101）或误动作。

- 变址修饰目标指定了超出软元件范围的范围时。
- 在未设置文件寄存器的状态下指定了文件寄存器时。

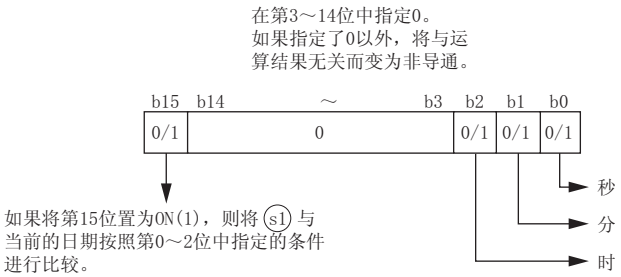
(2) 各项目的设置是以 BIN 值进行设置的。

(3) ⓢ1, ⓢ2 中的“时”是以 0～23(0 时～23 时）进行设置。（设置为 24 小时制。）

(4) ⓢ1 +1, ⓢ2 +1 中的“分”是以 0～59(0 分～59 分）进行设置。

(5) ⓢ1 +2, ⓢ2 +2 中的“秒”是以 0～59(0 秒～59 秒）进行设置。

(6) 通过在 n 中指定下图的值，可以对比较对象进行详细指定。
n 的位构成如下所示。



(a) 比较对象时间（第 0 ～ 2 位）

- 0：不对比较对象的时间数据（时 / 分 / 秒）进行比较。
- 1：对比较对象的时间数据（时 / 分 / 秒）进行比较。

(b) 比较运算对象（第 15 位）

- 0：进行 ⓢ1 中指定的时间数据与 ⓢ2 中指定的时间数据的比较。
- 1：进行 ⓢ1 中指定的时间数据与当前的时间数据的比较。
  ⓢ2 中指定的时间数据被忽略。

(c) 比较对象位的处理内容如下述所示。

与任意的时间数据比较时的 n 值	与当前的时间数据比较时的 n 值	比较对象时间	处理内容
0001H	8001H	秒	仅对秒（ⓢ1 +2）进行比较。
0002H	8002H	分	仅对分（ⓢ1 +1）进行比较。
0003H	8003H	分、秒	对分（ⓢ1 +1），秒（ⓢ1 +2）进行比较。
0004H	8004H	时	仅对时（ⓢ1）进行比较。
0005H	8005H	时、秒	对时（ⓢ1），秒（ⓢ1 +2）进行比较。
0006H	8006H	时、分	对时（ⓢ1），分（ⓢ1 +1）进行比较。
0007H	8007H	时、分、秒	对时（ⓢ1），分（ⓢ1 +1），秒（ⓢ1 +2） 全部进行比较。
0001H ～ 0007H, 8001H ～ 8007H 以外		无	不对时（ⓢ1），分（ⓢ1 +1），秒（ⓢ1 +2） 全部进行比较。（变为非导通状态。）

(7) 比较对象软元件中存储的数据不能被识别为时间数据的情况下，在指令执行后将 SM709 置为 ON 后，变为非导通状态。即使不能被识别为时间数据的情况下，只要在设置范围内 SM709 将不变为 ON。

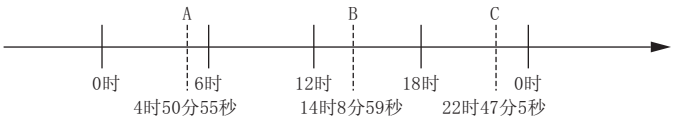
ⓢ1 ～ ⓢ1 +2 或 ⓢ2 ～ ⓢ2 +2 超出了指定软元件范围的情况下，将 SM709 置为 ON 后，变为非导通状态。

一旦将 SM709 置为 ON，在复位 / 电源 OFF 之前将保持为 ON 状态，因此应根据需要将其置为 OFF。

(8) 各指令的比较运算结果如下所示。

内指令符号	条件	比较运算结果	内指令符号	条件	比较运算结果
□TM=	ⓢ1 =ⓢ2	导通状态	□TM=	ⓢ1 ≠ⓢ2	非导通状态
□TM<>	ⓢ1 ≠ⓢ2		□TM<>	ⓢ1 =ⓢ2	
□TM>	ⓢ1 >ⓢ2		□TM>	ⓢ1 ≤ⓢ2	
□TM<=	ⓢ1 ≤ⓢ2		□TM<=	ⓢ1 >ⓢ2	
□TM<	ⓢ1 <ⓢ2		□TM<	ⓢ1 ≥ⓢ2	
□TM>=	ⓢ1 ≥ⓢ2		□TM>=	ⓢ1 <ⓢ2	

• 时间的比较示例如下所示。



上述所示的日期 A、B，C 的比较运算结果如下所示。

即使以相同条件进行了比较的情况下，根据选择的比较对象其比较运算结果将有所不同。

比较对象	比较条件		
	A<B	B<C	A<C
秒	○	×	×
分	×	○	×
分、秒	×	○	×
时	○	○	○
时、秒	○	○	○
时、分	○	○	○
时、分、秒	○	○	○
无	×	×	×

○：导通　×：非导通

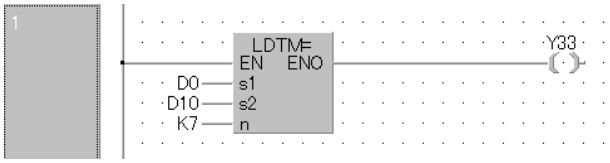
出 错

不存在 TM=、TM<>、TM>、TM<=、TM<、TM>= 指令相关的运算出错。

程序示例

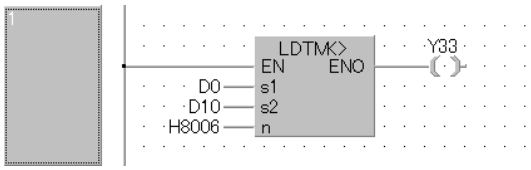
(1) 以下为将 D0 的数据与 D10 的数据（时、分、秒）进行比较，D0 的数据与 D10 的数据一致时将 Y33 置为 ON 的程序。

[结构体梯形图]



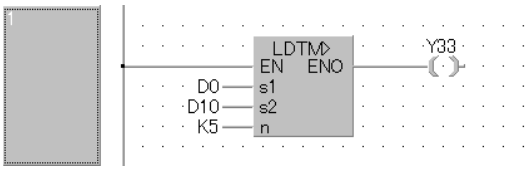
(2) 以下为将 D0 的数据与当前的时间数据（时、分）进行比较，D0 的数据与当前的时间数据不一致时将 Y33 置为 ON 的程序。

[结构体梯形图]



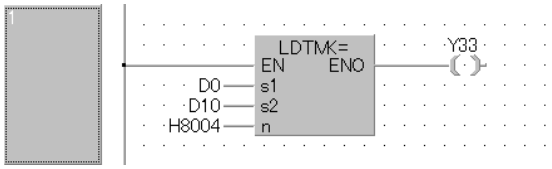
(3) 以下为将 D0 的数据与 D10 的数据（时、秒）进行比较，D10 的数据低于 D0 时将 Y33 置为 ON 的程序。

[结构体梯形图]



(4) 以下为将 D0 的数据与当前的时间数据（时）进行比较，当前的时间数据大于 D0 时将 Y33 置为 ON 的程序。

[结构体梯形图]



7.16 扩展时钟用指令

7.16.1 扩展时钟数据的读取

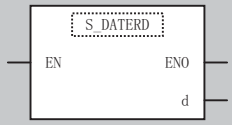
S(P)_DATERD

Basic × | Ver. High performance | Universal | L CPU

高性能型 QCPU：序列号的前 5 位数为 “07032”以后

S(P)_DATERD

结构体梯形图



ST

```
ENO:=  S_DATERD  (EN, d);
```

[ ] 中放入下述指令。

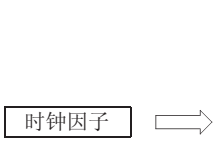
S_DATERD　　SP_DATERD

输入自变量， EN:　执行条件　：位
输出自变量， ENO:　执行结果　：位
d:　存储读取的时钟数据的软元件的起始编号　：ANY16(0..7)

设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
ⓓ	-	○		-					

功 能

(1) CPU 从 CPU 模块的时钟因子中读取 “年、月、日、时、分、秒、星期、1/1000 秒”，以 BIN 值存储到ⓓ中指定的软元件以后。



ⓓ	年(公历)	(1980～2079)
ⓓ+1	月	(1～12)
ⓓ+2	日	(1～31)
ⓓ+3	时(24小时制)	(0～23)
ⓓ+4	分	(0～59)
ⓓ+5	秒	(0～59)
ⓓ+6	星期	(0～6)
ⓓ+7	1/1000秒	(0～999)

(2) ⓓ中的 “年”是以公历 4 位进行存储。

(3) ⓓ+6 中的 “星期” 的 “周日～周六” 是以 “0 ～ 6” 进行存储。

星期	日	一	二	三	四	五	六
存储数据	0	1	2	3	4	5	6

(4) 闰年将被自动修正。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

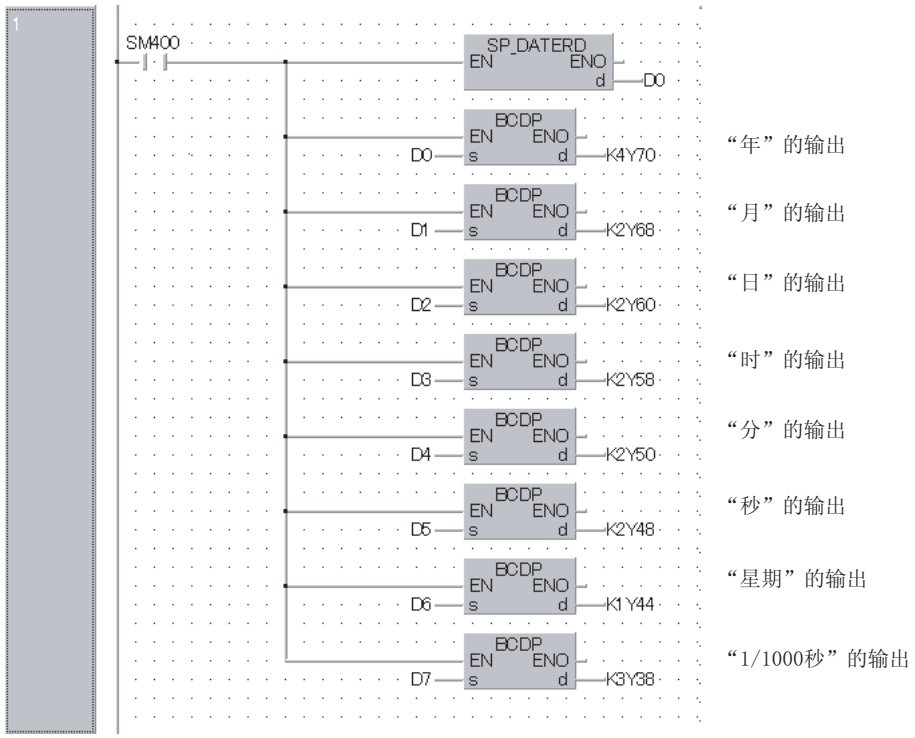
- ⓓ中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时） （出错代码： 4101）

程序示例

(1) 以下为将下述时钟数据以 BCD 值进行输出的程序。

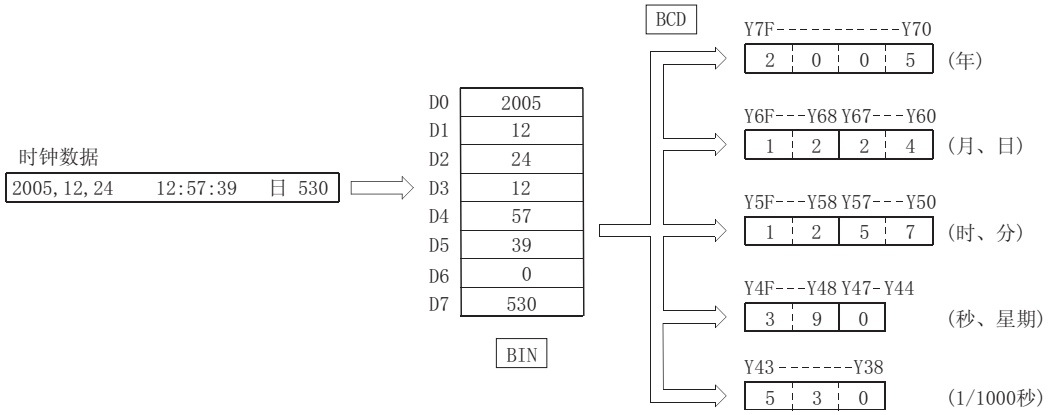
年 ..... Y70 ～ Y7F
月 ..... Y68 ～ Y6F
日 ..... Y60 ～ Y67
时 ..... Y58 ～ Y5F
分 ..... Y50 ～ Y57
秒 ..... Y48 ～ Y4F
星期 ..... Y44 ～ Y47
1/1000 秒 ..... Y38 ～ Y43

[结构体梯形图]



```
[ST]
IF SM400 THEN
     SP_DATERD(TRUE,D0);
     BCDP(TRUE,D0,K4Y70);
     BCDP(TRUE,D1,K2Y68);
     BCDP(TRUE,D2,K2Y60);
     BCDP(TRUE,D3,K2Y58);
     BCDP(TRUE,D4,K2Y50);
     BCDP(TRUE,D5,K2Y48);
     BCDP(TRUE,D6,K1Y44);
     BCDP(TRUE,D7,K3Y738);
END_IF;
```

[动作]



注意事项

(1) 执行本指令时，即使 CPU 模块中设置了错误的时钟数据的情况下，也将读取时钟数据并存储到软元件中。(例： 2 月 30 日)

在 DATEWR 指令及 GX Works2 中设置时钟数据时，应设置正确的时钟数据。

(2) 读取 1/1000 秒的时钟数据时的误差最大为 2ms。(CPU 模块内部的时钟因子中存储的数据与通过本指令读取的数据之间的误差。)

(3) 进行位软元件的位数指定时只有在满足下述 (a)、(b) 的条件的情况下才可以使用。

(a) 位数的指定： K4

(b) 软元件的起始： 16 的倍数

在未满足上述 (a)、(b) 的条件的情况下，将变为 INSTRCT CODE ERR.(出错代码： 4004) 状态。

7.16.2 扩展时钟数据的加法

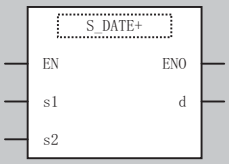
S(P)_DATE+

Basic | High performance Ver. | Universal | L CPU

高性能型 QCPU：序列号的前 5 位数为 “07032”以后

S(P)_DATE+

结构体梯形图



ST

不支持。

□ 中放入下述指令。

S_DATE+　　SP_DATE+

输入自变量，EN:　执行条件　：位

s1:　存储被进行加法运算的时间数据的软元件的起始编号　：ANY16

s2:　存储进行加法运算的时间（时间）数据的软元件的起始编号　：ANY16

输出自变量，ENO:　执行结果　：位

d:　存储加法运算结果时间（时间）数据的软元件的起始编号　：ANY16

设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
ⓢ1	–	○		–					
ⓢ2	–	○		–					
ⓓ	–	○		–					

功 能

(1) 将ⓢ1中指定的时间数据与ⓢ2中指定的时间数据进行加法运算，将加法运算结果存储到ⓓ中指定的软元件编号以后。

		设置数据			设置数据			设置数据
ⓢ1	时	(0～23)	ⓢ2	时	(0～23)	ⓓ	时	(0～23)
ⓢ1+1	分	(0～59)	ⓢ2+1	分	(0～59)	ⓓ+1	分	(0～59)
ⓢ1+2	秒	(0～59) +	ⓢ2+2	秒	(0～59) ⇒	ⓓ+2	秒	(0～59)
ⓢ1+3	–		ⓢ2+3	–		ⓓ+3	–	
ⓢ1+4	1/1000秒	(0～999)	ⓢ2+4	1/1000秒	(0～999)	ⓓ+4	1/1000秒	(0～999)

例如，将 6 时 32 分 40 秒 875 与 7 时 48 分 10 秒 500 进行加法运算的情况如下所示。

ⓢ1	6时		ⓢ2	7时		ⓓ	14时
ⓢ1 +1	32分		ⓢ2 +1	48分		ⓓ +1	20分
ⓢ1 +2	40秒	+	ⓢ2 +2	10秒	⇨	ⓓ +2	51秒
ⓢ1 +3	－		ⓢ2 +3	－		ⓓ +3	－
ⓢ1 +4	875		ⓢ2 +4	500		ⓓ +4	375

(2) 运算结果时间超过了 24 时的情况下，将减去 24 小时的值作为运算结果。

例如，将 14 时 20 分 30 秒 875 与 20 时 20 分 20 秒 500 进行加法运算的情况下，其结果不是 34 时 40 分 51 秒 375，而是 10 时 40 分 51 秒 375。

ⓢ1	14时		ⓢ2	20时		ⓓ	10时
ⓢ1 +1	20分		ⓢ2 +1	20分		ⓓ +1	40分
ⓢ1 +2	30秒	+	ⓢ2 +2	20秒	⇨	ⓓ +2	51秒
ⓢ1 +3	－		ⓢ2 +3	－		ⓓ +3	－
ⓢ1 +4	875		ⓢ2 +4	500		ⓓ +4	375

要 点

在运算中不使用 ⓢ1 +3, ⓢ2 +3, ⓓ +3 的软元件。

通过 S(P)_DATERD 读取的时钟数据可以直接进行加法运算。

ⓓ	时
ⓓ +1	分
ⓓ +2	秒
ⓓ +3	星期
ⓓ +4	1/1000秒

通过S(P)_DATERD指令进行了读取的情况下，在秒与1/1000秒之间加入星期。
在S(P)_DATE+指令中不进行运算，因此可以直接进行加法运算。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1, ⓢ2 的设置数据超出了范围时。(功能 (1)) (出错代码 : 4100)
- ⓢ1, ⓢ2, ⓓ 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码 : 4101)

注意事项

进行位软元件的位数指定时只有在满足下述 (a)、(b) 的条件的情况下才可以使用。

(a) 位数的指定 : K4

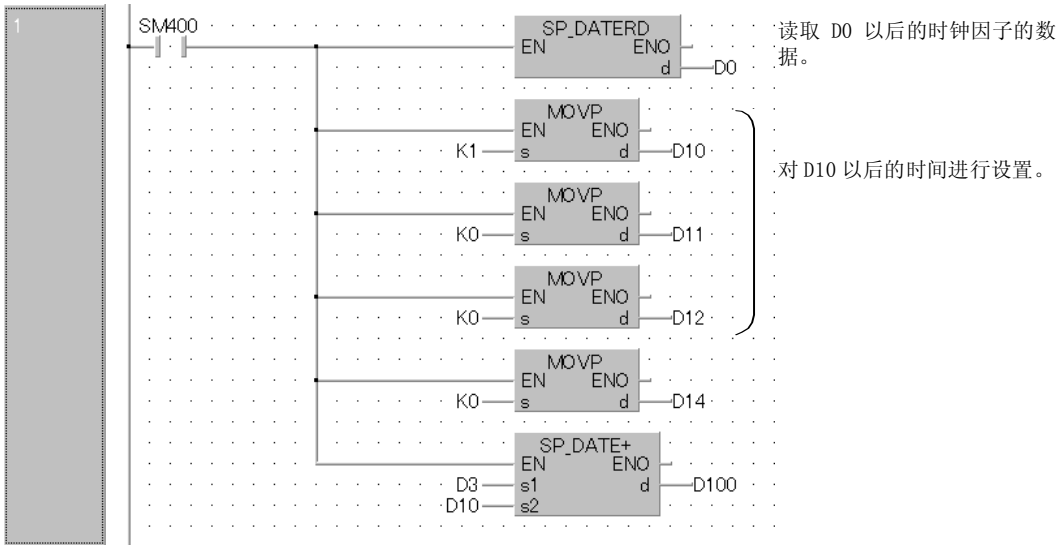
(b) 软元件的起始 : 16 的倍数

在未满足上述 (a)、(b) 的条件的情况下，将变为 INSTRCT CODE ERR.(出错代码 : 4004) 状态。

程序示例

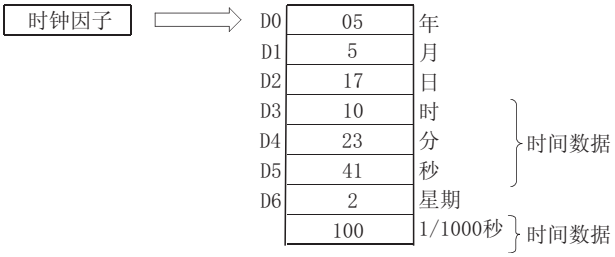
(1) 以下为 X20 变为 ON 时，将从时钟因子中读取的时间数据与 1 小时进行加法运算，将运算结果存储到 D100 以后的程序。

[结构体梯形图]

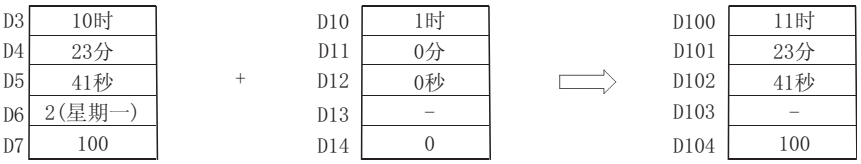


[动作]

• 通过 SP_DATERD 指令读取时间数据



• 通过 SP_DATE + 指令进行加法运算



7.16.3 扩展时钟数据的减法

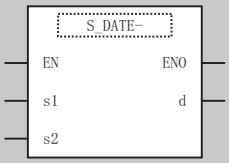
S(P)_DATE-

Basic | Ver. High performance | Universal | L CPU

高性能型 QCPU：序列号的前 5 位数为 “07032”以后

S(P)_DATE-

结构体梯形图



ST

不支持。

中放入下述指令。

S_DATE-　　SP_DATE-

输入自变量，EN:　执行条件　：位
输出自变量，ENO:　执行结果　：位
s1:　存储被进行减法运算的时间数据的软元件的起始编号：ANY16
s2:　存储减法运算时间（时间）数据的软元件的起始编号：ANY16
d:　存储减法结果时间（时间）数据的软元件的起始编号：ANY16

设置数据	内部软元件		Z, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
ⓢ1	-	○		-					
ⓢ2	-	○		-					
ⓓ	-	○		-					

功 能

(1) 将ⓢ1中指定的时间数据与ⓢ2中指定的时间数据进行减法运算，将减法运算结果存储到ⓓ中指定的软元件编号以后。

		设置数据			设置数据			设置数据
ⓢ1	时	(0～23)	ⓢ2	时	(0～23)	ⓓ	时	(0～23)
ⓢ1+1	分	(0～59)	ⓢ2+1	分	(0～59)	ⓓ+1	分	(0～59)
ⓢ1+2	秒	(0～59)	ⓢ2+2	秒	(0～59)	ⓓ+2	秒	(0～59)
ⓢ1+3	-		ⓢ2+3	-		ⓓ+3	-	
ⓢ1+4	1/1000秒	(0～999)	ⓢ2+4	1/1000秒	(0～999)	ⓓ+4	1/1000秒	(0～999)

－ ⇨

例如，从 10 时 40 分 20 秒 875 中减去 3 时 50 分 10 秒 500 的情况如下所示。

(s1)	10时		(s2)	3时		(d)	6时
(s1)+1	40分		(s2)+1	50分		(d)+1	50分
(s1)+2	20秒	－	(s2)+2	10秒	⇨	(d)+2	10秒
(s1)+3	－		(s2)+3	－		(d)+3	－
(s1)+4	875		(s2)+4	500		(d)+4	375

(2) 运算结果时间变为负数的情况下，将该数据进行 +24 后的值将成为运算结果。

例如，从 4 时 50 分 32 秒 875 中减去 10 时 42 分 12 秒 500 的情况下，其结果不是－ 6 时 8 分 20 秒 375，而是 18 时 8 分 20 秒 375。

(s1)	4时		(s2)	10时		(d)	18时
(s1)+1	50分		(s2)+1	42分		(d)+1	8分
(s1)+2	32秒	－	(s2)+2	12秒	⇨	(d)+2	20秒
(s1)+3	－		(s2)+3	－		(d)+3	－
(s1)+4	875		(s2)+4	500		(d)+4	375

要 点

在运算中不使用 (s1)+3，(s2)+3，(d)+3 的软元件。

通过 S(P)_DATERD 读取的时钟数据可以直接进行减法运算。

(d)	时
(d)+1	分
(d)+2	秒
(d)+3	星期
(d)+4	1/1000秒

通过S(P)_DATERD指令进行了读取的情况下，在秒与1/1000秒之间加入星期。
在S(P)_DATE-指令中不进行运算，因此可以直接进行减法运算。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- (s1)，(s2) 的设置数据超出了范围时。(功能 (1)) (出错代码： 4100)
- (s1)，(s2)，(d) 中指定的软元件超出了相应软元件的范围时。
  (通用型 QCPU、LCPU 时) (出错代码： 4101)

注意事项

进行位软元件的位数指定时只有在满足下述 (a)、(b) 的条件的情况下才可以使用。

(a) 位数的指定： K4

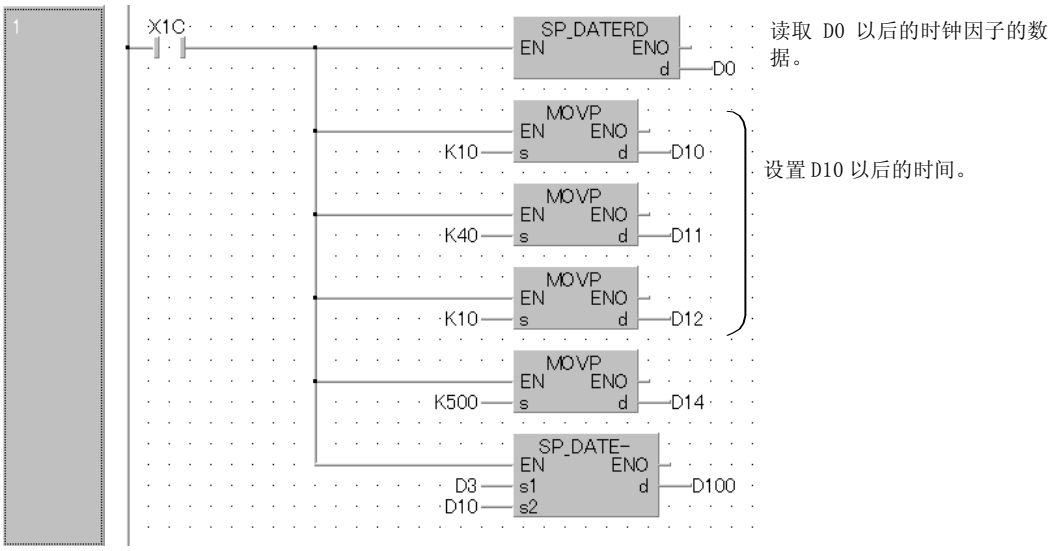
(b) 软元件的起始 ： 16 的倍数

在未满足上述 (a)、(b) 的条件的情况下，将变为 INSTRCT CODE ERR.(出错代码： 4004) 状态。

程序示例

(1) 以下为 X1C 变为 ON 时，将从时钟因子中读取的时间数据与 D10 以后中存储的时间数据进行减法运算后，将运算结果存储到 D100 以后的程序。

[结构体梯形图]



[动作]

• 通过 SP_DATERD 指令读取时间数据。

时钟因子 ⇨

D0	05	年
D1	2	月
D2	23	日
D3	8	时
D4	42	分 } 时间数据
D5	1	秒
D6	3	星期
D7	997	1/1000秒 } 时间数据

• 通过 SP_DATE- 指令进行减法运算。

D3	8时		D10	10时	⇨	D100	22时
D4	42分		D11	40分		D101	1分
D5	1秒	−	D12	10秒		D102	51秒
D6	3(星期三)		D13	−		D103	−
D7	997		D14	500		D104	497

8时42分1秒997－10时40分10秒500 ⇨ －2时1分51秒497

↓ 加上24

⇩

22时1分51秒497

7.17 程序控制用指令

7.17.1 程序待机

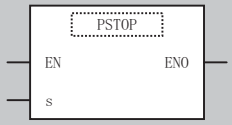
PSTOP

Basic | High performance | Universal | L CPU

PSTOP(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= PSTOP (EN, s);

[ ] 中放入下述指令。

PSTOP PSTOPP

输入自变量, EN: 执行条件 : 位
s: 置为待机状态的程序的文件名的字符串数据 : 字符串
输出自变量, ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
Ⓢ	-	○		-				○	-

功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为待机状态。

(2) 只有驱动器 0(程序存储器 / 内置 RAM) 中存储的程序才可以被置为待机类型。

(3) 指定的程序在 END 处理时变为待机状态。

(4) 即使在参数中指定为执行形式的情况下本指令也将优先。

(5) 文件名中不需要指定扩展名 (.QPG)。
(仅以 .QPG 的文件作为对象。)

出 错

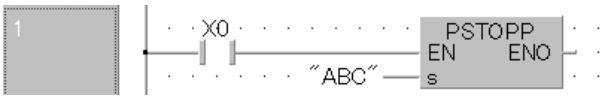
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定的文件名的程序不存在时。 (出错代码： 2410)
- ⓢ的文件名存储目标软元件超出了相应软元件的范围时。 (出错代码： 4101)

程序示例

以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为待机状态的程序。

[结构体梯形图]



[ST]

```
PSTOPP(X0,"ABC");
```

7.17.2 程序输出 OFF 待机

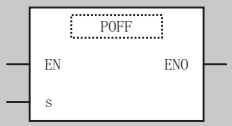
POFF

Basic | High performance | Universal | L CPU

POFF(P)

P: 执行条件 : ⏎

结构体梯形图



ST

ENO:= POFF (EN, s);

[ ] 中放入下述指令。

POFF POFFP

输入自变量， EN: 执行条件 : 位
s: 将输出置为 OFF 后置为待机类型的程序的文件名 : 字符串
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
Ⓢ	-	○		-				○	-

功 能

(1) 对Ⓢ中指定的软元件中存储的文件名的程序的执行类型进行变更。

- 扫描执行类型 : 在下一个扫描中将输出置为 OFF(非执行处理)。在下下一个扫描以后，变为待机类型。
- 低速执行类型 : 使低速执行类型的执行中断，在下一个扫描中将输出置为 OFF。在下下一个扫描以后，变为待机类型。

(2) 只有驱动器 0(程序存储器 / 内置 RAM) 中存储的程序才可以置为待机类型。

(3) 即使在参数中指定为执行形式的情况下本指令也将优先。

(4) 文件名中不需要指定扩展名 (.QPG)。
(仅以 .QPG 的文件作为对象。)

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定的文件名的程序不存在时。 （出错代码： 2410）
- ⓢ的文件名存储目标软元件超出了相应软元件的范围时。 （出错代码： 4101）

备 注

1. 非执行处理是指，对各线圈指令进行与条件设置为 OFF 状态时相同的处理。

2. 非执行处理后的各线圈指令的运算结果与条件触点的 ON/OFF 状态无关，其情况如下所示。

- OUT 指令..... 强制 OFF
- SET 指令
- RST 指令
- SFT 指令
- 基本指令
- 应用指令

... 保持状态

- PLS 指令
- 脉冲化指令 (□ P)

... 与条件触点 OFF 相同的处理

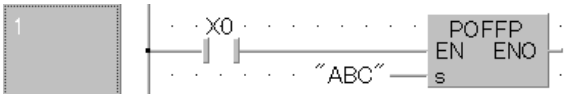
- 低速 / 高速定时器的当前值 0
- 累计定时器的当前值
- 计数器的当前值

... 保持

程序示例

以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为非执行，进入待机状态的程序。

[结构体梯形图]



[ST]

```
POFFP(X0,"ABC");
```

7.17.3 程序扫描执行登录

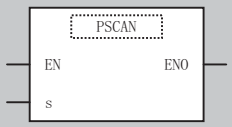
PSCAN

Basic　High performance　Universal　L CPU

PSCAN(P)

P: 执行条件　　: ↰

结构体梯形图



ST

ENO:= PSCAN (EN, s);

[ ] 中放入下述指令。

PSCAN　　PSCANP

输入自变量， EN:　执行条件　　: 位
　　s:　置为扫描执行类型的程序的文件名　　: 字符串
输出自变量， ENO:　执行结果　　: 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
Ⓢ	-	○		-				○	-

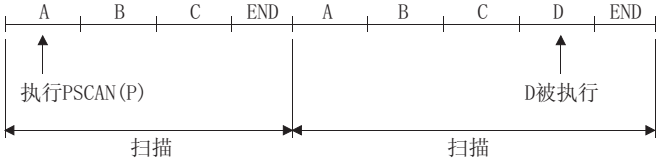
功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为扫描执行类型。

(2) 只有驱动器 0(程序存储器 / 内置 RAM) 中存储的程序才可以被置为扫描执行类型。

(3) 指定的程序在 END 处理时变为执行类型。

例 存在有程序 A、B、C 时，在 A 程序中对 D 程序执行了 “PSCAN(P)” 的情况。



(4) 即使在参数中指定为执行形式的情况下本指令也将优先。

(5) 文件名中不需要指定扩展名 (.QPG)。
(仅以 .QPG 的文件作为对象。)

出 错

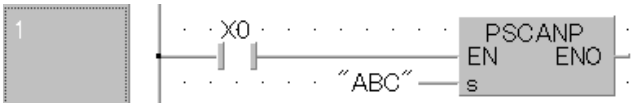
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定的文件名的程序不存在时。 （出错代码： 2410）
- ⓢ的文件名存储目标软元件超出了相应软元件的范围时。 （出错代码： 4101）
- 指定的文件名为 SFC 程序，已有其它文件名的 SFC 程序启动时。
  (SFC 程序的双重启动出错)
  ((通用型 QCPU、LCPU 时) （出错代码： 4131）

程序示例

以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为扫描执行类型的程序。

[结构体梯形图]



[ST]

```
PSCANP(X0,"ABC");
```

7.17.4 程序低速执行登录

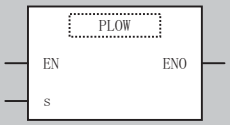
PLOW

Basic　High performance　Universal　L CPU

PLOW(P)

P: 执行条件　　: ⊥

结构体梯形图



ST

ENO:= PLOW (EN, s);

□ 中放入下述指令。

PLOW　　PLOWP

输入自变量， EN:　执行条件　　: 位
　　s:　置为低速执行类型的程序的文件名　　: 字符串
输出自变量， ENO:　执行结果　　: 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
Ⓢ	-	○		-				○	-

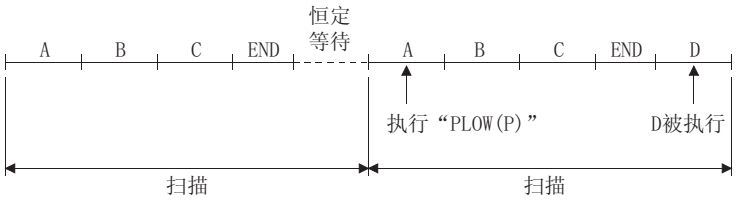
功 能

(1) 将Ⓢ中指定的软元件中存储的文件名的程序置为低速执行类型。

(2) 只有驱动器 0(程序存储器 / 内置 RAM) 中存储的程序才可以置为低速执行类型。

(3) 指定的程序在 END 处理中变为低速执行类型。

例 存在有程序 A、B、C 时，在 A 程序中将对 D 程序执行了 “PLOW(P)” 的情况下。
(假设已进行了恒定扫描的设置。)



(4) 即使在参数中指定为执行形式的情况下本指令也将优先。

(5) 文件名中不需要指定扩展名 (.QPG)。
(仅以 .QPG 的文件作为对象。)

出 错

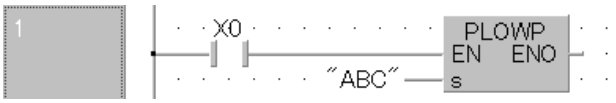
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定的文件名的程序不存在时。 (出错代码： 2410)
- 指定的文件名的程序中存在有 CHK 指令时。 (出错代码： 4235)

程序示例

以下为 X0 变为 ON 时，将文件名为 ABC 的程序置为低速执行类型的程序。

[结构体梯形图]



[ST]

```
PLOWP(X0,"ABC");
```

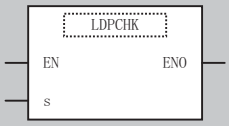
7.17.5 程序执行状态检查

LDPCHK, ANDPCHK, ORPCHK

Basic | High performance | Universal | L CPU

LDPCHK
ANDPCHK
ORPCHK

结构体梯形图



ST

```
ENO:= LDPCHK (EN, s);
```

[ ] 中放入下述指令。

LDPCHK　　ANDPCHK
ORPCHK

输入自变量，EN:　执行条件　：位
　　s:　检查执行状态的程序的文件名　：字符串
输出自变量，ENO:　执行结果　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
Ⓢ			-					○	-

功 能

(1) 对指定的文件名的程序是处于执行中状态还是不处于执行中状态（非执行）进行检查。

(2) 指定的文件名的程序处于执行中时，导通，非执行中时变为非导通。

(3) 文件名中指定去掉了扩展名（.QPG）的文件名。
例如文件名为 ABC.QPG 的情况下，指定为 “ABC”。

出 错

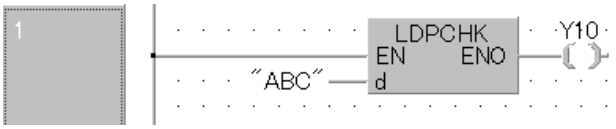
在以下情况下将变为运算出错状态，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定的文件名的程序不存在时。　（出错代码：2410）

程序示例

以下为程序文件 “ABC.QPG”处于执行中时，将 Y10 置为 ON 的程序。

[结构体梯形图]



[ST]

```
OUT(LDPCHK(TRUE,"ABC"),Y10);
```

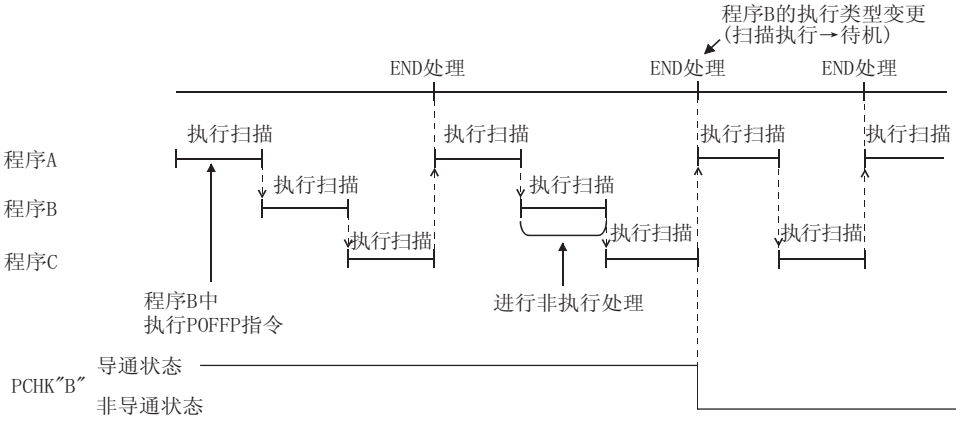
备 注

非执行中表示程序的执行类型处于待机类型的状态。
执行中表示程序的执行类型为扫描执行类型（包括输出 OFF 中（非执行处理中））、低速执行类型、恒定周期执行类型的状态。

要 点

对于 PCHK 指令，指定的文件名的程序（对象程序）为执行中时变为导通状态，为非执行中时变为非导通状态。
通过 POFF(P) 指令将对象程序置为非执行（待机类型）的情况下，在进行对象程序的非执行处理期间，PCHK 指令变为导通状态。
在非执行处理结束的扫描的 END 处理中，对象程序将变为非执行（待机类型），PCHK 指令将变为非导通状态。
因此，对于通过 POFF(P) 指令结束了非执行处理的程序，如果执行 PCHK 指令，PCHK 指令将可能变为导通状态，因此应加以注意。

以程序 A →程序 B →程序 C 的顺序执行的情况下，程序 A 执行程序 B 的 POFF(P) 指令，程序 C 执行程序 B 的 PCHK 指令时的动作如下图所示。



7.18 其它指令

7.18.1 WDT 复位

WDT

Basic | High performance | Universal | L CPU

WDT(P)

P: 执行条件 :

结构体梯形图

ST

ENO:= WDT (EN);

中放入下述指令。

WDT WDTP

输入自变量，EN: 执行条件 : 位
输出自变量，ENO: 执行结果 : 位

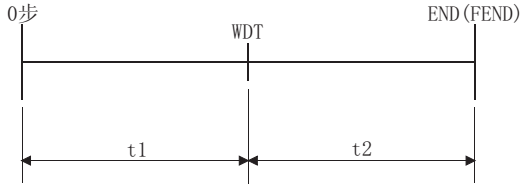
设置数据	内部软元件		R，ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
-	-								

功 能

(1) 在顺控程序中对看门狗定时器进行复位。

(2) 在扫描时间根据条件超过了看门狗定时器的设置值的情况下使用。
在扫描时间超过了各个扫描的看门狗定时器的设置值的情况下，应在编程工具的参数设置中，对看门狗定时器的设置值进行变更。

(3) 在对 0 步开始至 WDT(P) 指令为止的 t1，以及 WDT(P) 指令开始至 END(FEND) 指令为止的 t2 进行设置时，应不超过看门狗定时器的设置值。



(4) WDT(P) 指令在 1 个扫描中可以使用 2 次以上，但至发生异常时的输出 OFF 需要耗费一定的时间，应加以注意。

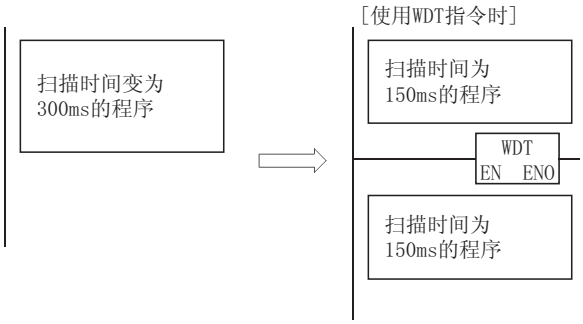
(5) 即使执行了 WDT(P) 指令，特殊寄存器中存储的扫描时间的值也不会被清除。
因此，各特殊寄存器的扫描时间值有可能会大于参数中设置的看门狗定时器的设置值。

出 错

不存在 WDT(P) 指令相关的运算出错。

程序示例

以下为看门狗定时器的设置为 200ms，根据程序的执行条件 0 ～ END(FEND) 指令的时间为 300ms 时的程序。

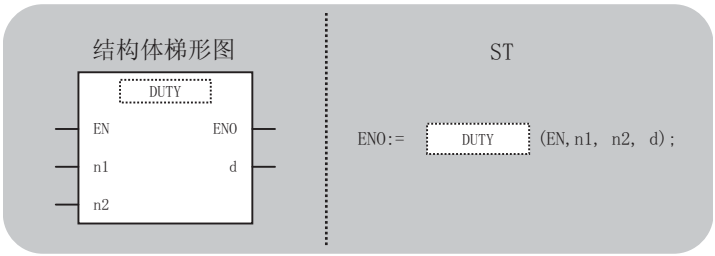


7.18.2 定时脉冲发生

DUTY

Basic | High performance | Universal | L CPU

DUTY



[ ] 中放入下述指令。

DUTY

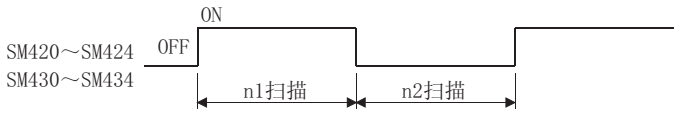
输入自变量， EN: 执行条件 : 位
n1: 置为 ON 的扫描数 : ANY16
n2: 置为 OFF 的扫描数 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 用户用计时时钟 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它数据
	位	字		位	字				
n1	○	○							-
n2	○	○							-
ⓓ	○*1	-							-

*1 : 只能使用 SM420 ～ SM424、SM430 ～ SM434。

功 能

(1) 将ⓓ中指定的用户用计时时钟 (SM420 ～ SM424、SM430 ～ SM434) 按 n1 中指定的扫描数置为 ON，按 n2 中指定的扫描数置为 OFF。



(2) 在扫描执行类型程序中使用 SM420 ～ SM424，在低速执行类型程序中使用 SM430 ～ SM434。

(3) n1、n2 被设置为 0 时的情况如下所示。

(a) n1=0, n2≧0 ... SM420 ～ SM424/SM430 ～ SM434 保持为 OFF 状态不变。

(a) n1＞0, n2=0 .... SM420 ～ SM424/SM430 ～ SM434 保持为 ON 状态不变。

(4) 执行 DUTY 指令时将 n1、n2、ⓓ中指定的数据登录到系统中，在 END 处理中进行定时脉冲的 ON/OFF。

出 错

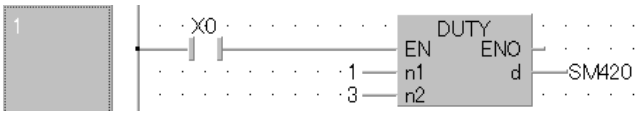
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 中指定的软元件超出了 SM420 ～ SM424、SM430 ～ SM434 的范围时。
  （出错代码： 4101）
- n1、n2 低于 0 时。 （出错代码： 4100）

程序示例

以下为 X0 变为 ON 时，将 SM420 进行 1 个扫描 ON 后，进行 3 个扫描 OFF 的程序。

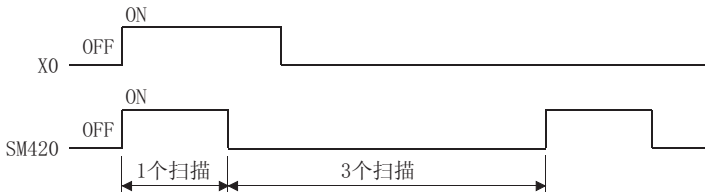
[结构体梯形图]



[ST]

```
DUTY(X0,1,3,SM420);
```

[动作]



7.18.3 时间检查

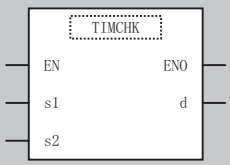
TIMCHK

Ver.
Basic | High performance | Universal | L CPU

基本型 QCPU：序列号的前 5 位数为 “04122”以后

TIMCHK

结构体梯形图



ST

ENO:= TIMCHK (EN, s1, s2, d);

[ ] 中放入下述指令。

TIMCHK

输入自变量， EN: 执行条件 : 位
s1: 存储计测的当前值的软元件 : ANY16
s2: 存储计测的设置值的软元件 : ANY16
输出自变量， ENO: 执行结果 : 位
d: 时间到时置为 ON 的软元件 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
ⓢ1	-	○		-					-
ⓢ2	○	○		○					-
ⓓ	○	-		-					-

功 能

(1) 对条件软元件的 ON 时间进行计测，如果连续 ON 的时间超过了 ⓢ2 中指定的软元件中设置的时间，则将 ⓓ 中指定的软元件置为 ON。

(2) ⓢ1 中指定的软元件的当前值的清零及 ⓓ 中指定的软元件的 OFF 是在执行指令的上升沿时进行。

对于 ⓢ1 中指定的软元件的当前值与 ⓓ 中指定的软元件的 ON 状态，即使执行指令变为 OFF 时其状态也仍将被保持。

(3) 计测的设置值是以 100ms 为单位进行设置。

出 错

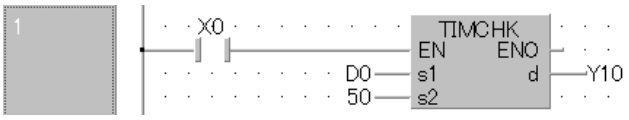
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定了不能指定的软元件时。 （出错代码： 4100）

程序示例

以下为将 X0 的 ON 时间的设置值设置为 5 秒，将当前值存储软元件设置为 D0，将时间到时置为 ON 的软元件设置为 Y10 的程序。

[结构体梯形图]



[ST]

```
TIMCHK(X0,D0,50,Y10);
```

7.18.4 文件寄存器直接 1 字节读取

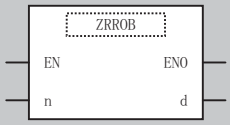
ZRRDB

Basic | High performance | Universal | L CPU

ZRRDB(P)

P: 执行条件 : ↰

结构体梯形图



ST

```
ENO:= ZRROB (EN, n, d);
```

□ 中放入下述指令。

ZRRDB ZRRDBP

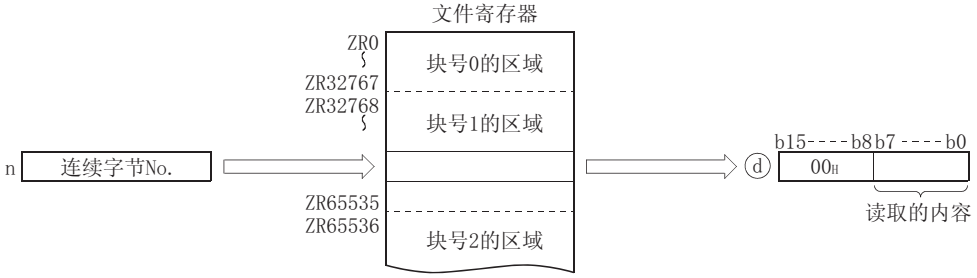
输入自变量， EN: 执行条件 : 位
n: 读取的文件寄存器的连续字节 No. : ANY32
输出自变量， ENO: 执行结果 : 位
d: 存储被读取的数据的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
n			○					○	-
ⓓ			○					-	-

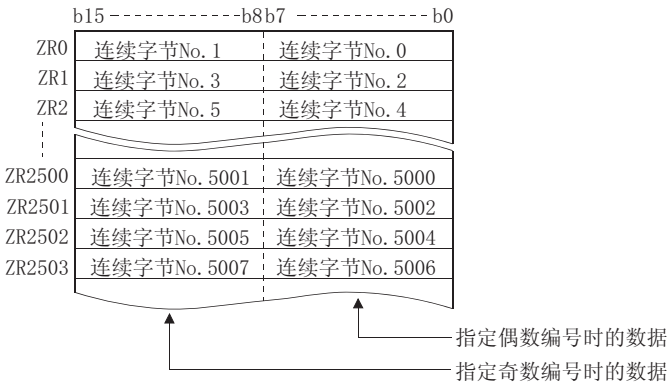
功 能

(1) 在忽略块号的状况下，对 n 中指定的连续字节 No. 的文件寄存器的内容进行读取后，存储到ⓓ中指定的软元件的低 8 位中。

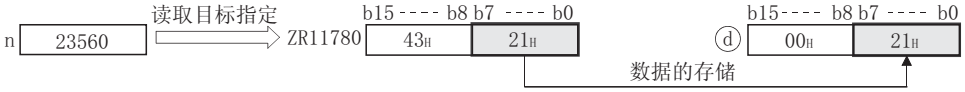
ⓓ中指定的高 8 位将变为 00H。



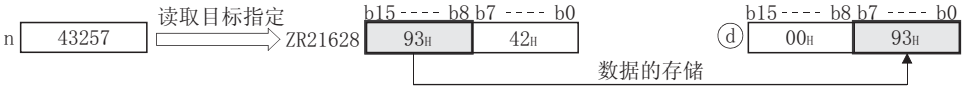
(2) 对应于连续字节 No. 的文件寄存器的编号如下所示。



(a) 指定为 n=23560 的情况下，对 ZR11780 的低 8 位的数据进行读取。



(b) 指定为 n=43257 的情况下，对 ZR21628 的高 8 位的数据进行读取。



出 错

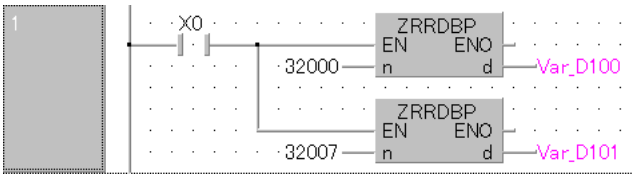
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定了超出允许指定范围的软元件编号（连续字节 No.）时。 （出错代码： 4101）

程序示例

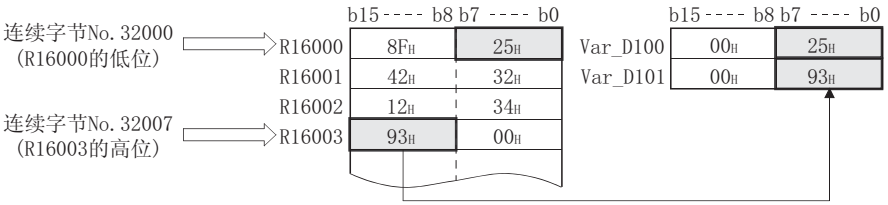
以下为 X0 变为 ON 时，对 ZR16000 的低位及 R16003 的高位进行读取后，存储到 Var_D100、Var_D101 中的程序。

[结构体梯形图]



```
[ST]
IF X0 THEN
     ZRRDBP(TRUE,32000,Var_D100);
     ZRRDBP(TRUE,32007,Var_D101);
END_IF;
```

[动作]



7.18.5 文件寄存器直接 1 字节写入

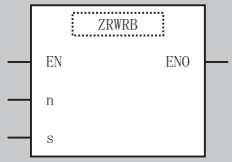
ZRWRB

Basic | High performance | Universal | L CPU

ZRWRB(P)

P: 执行条件 : ↑

结构体梯形图



ST

```
ENO:= ZRWRB (EN, n, s);
```

[ ] 中放入下述指令。

ZRWRB ZRWRBP

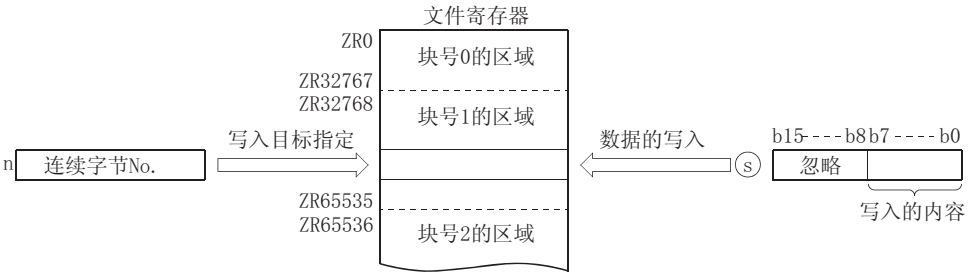
输入自变量， EN: 执行条件 : 位
n: 写入文件寄存器的连续字节 No. : ANY32
s: 存储写入数据的软元件的编号 : ANY16
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
n		○							-
ⓢ		○							-

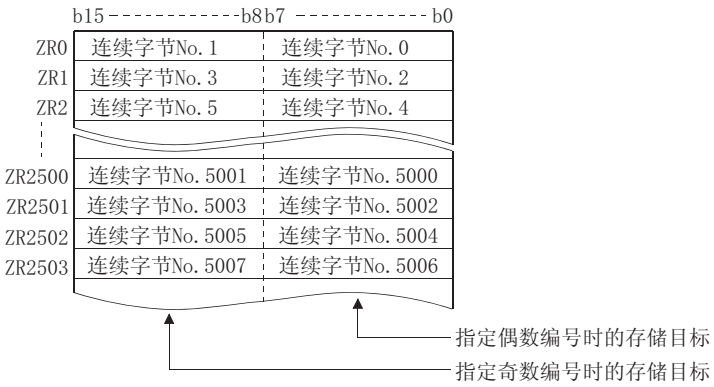
功 能

(1) 在忽略块号的状况下，将ⓢ中指定的软元件中存储的低位的内容，写入到 n 中指定的连续字节 No. 的文件寄存器中。

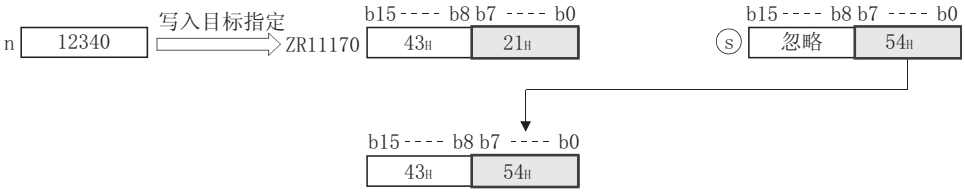
ⓢ中指定的软元件的高 8 位的数据将被忽略。



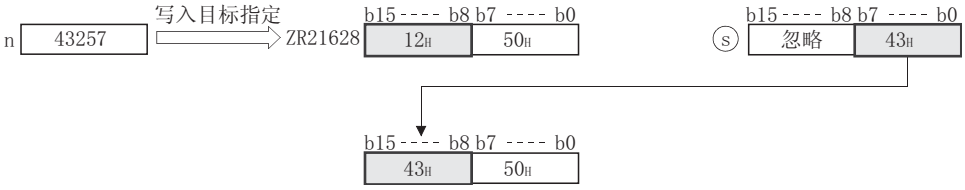
(2) 对应于连续字节 No. 的文件寄存器的编号如下所示。



指定为 n=12340 的情况下，写入到 ZR11170 的低 8 位中。



指定为 n=43257 的情况下，写入到 ZR21628 的高 8 位中。



出 错

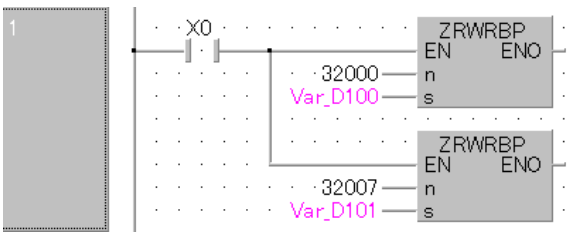
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 指定了超出允许指定范围的软元件编号（连续字节 No.）时。 （出错代码： 4101）

程序示例

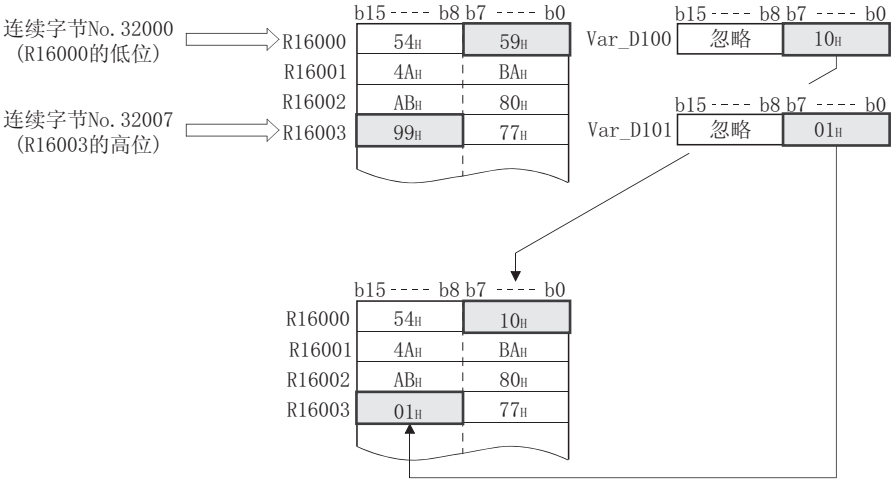
以下为 X0 变为 ON 时，将 Var_D100、Var_D101 的低位的数据写入到 ZR16000 的低位及 R16003 的高位中的程序。

[结构体梯形图]



```
[ST]
IF X0 THEN
     ZRWRBP(TRUE,32000,Var_D100);
     ZRWRBP(TRUE,32007,Var_D101);
END_IF;
```

[动作]



7.18.6 间接地址读取

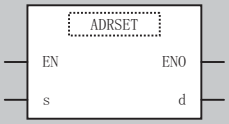
ADRSET

Basic | High performance | Universal | L CPU

ADRSET(P)

P: 执行条件 : ↰

结构体梯形图



ST

```
ENO:= ADRSET (EN, s, d);
```

□ 中放入下述指令。

ADRSET ADRSETP

输入自变量， EN: 执行条件 : 位
s: 间接地址 : ANY_SIMPLE
输出自变量， ENO: 执行结果 : 位
d: s 中指定的软元件的间接地址 : ANY32

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
ⓢ		○			-				
ⓓ		○			-				

功 能

(1) 将ⓢ中指定的软元件的间接地址存储到ⓓ的软元件中。

在顺控程序中执行软元件的间接地址的情况下，使用ⓓ中指定的软元件中存储的地址。

将D0的间接地址存储到D100中。

(2) ⓢ中不能进行位软元件的位数指定。

出 错

不存在 ADRSET(P) 指令相关的运算出错。

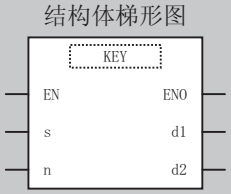
7.18.7 从键盘的数字键输入

KEY

Basic　High performance　Universal　L CPU

KEY

结构体梯形图



ST

ENO:= KEY (EN, s, n, d1, d2);

[ ] 中放入下述指令。

KEY

输入自变量，EN:　执行条件　：位

s:　进行数字输入的 (X) 的软元件　：位的数组 (0..8)

n:　输入数字的位数　：ANY16

输出自变量，ENO:　执行结果　：位

d1:　输入后的数字　：ANY16 的数组 (0..2)

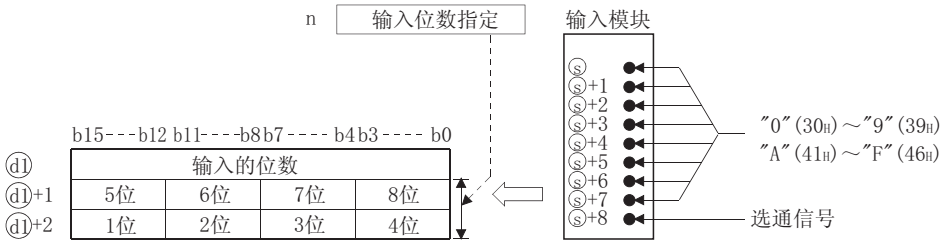
d2:　输入结束时置为 ON 的位软元件编号　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它数据
	位	字		位	字				
ⓢ	○(仅X)*1	-		-			-		-
n	○	○		○			○		-
ⓓ1	-	○		-			-		-
ⓓ2	○	○		○			-		-

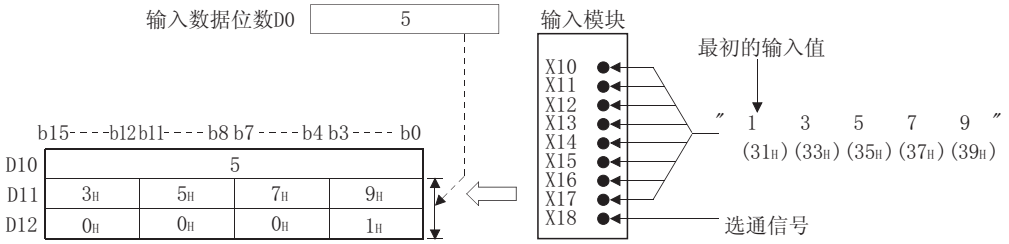
*1: 在全局标签中，应指定在软元件中设置了 X 的数组。

功 能

(1) 获取ⓢ中指定的输入 (X) 的 8 点的 ASCII 数据，转换为 16 进制数值后存储到ⓓ1中指定的软元件以后。

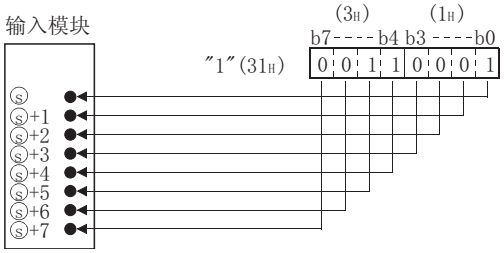


例如将输入数据的位数 (n) 设置为 5，在输入模块的 X10 ～ X18 中输入了 “31”、“33”、“35”、“37”、“39” 的情况如下所示。



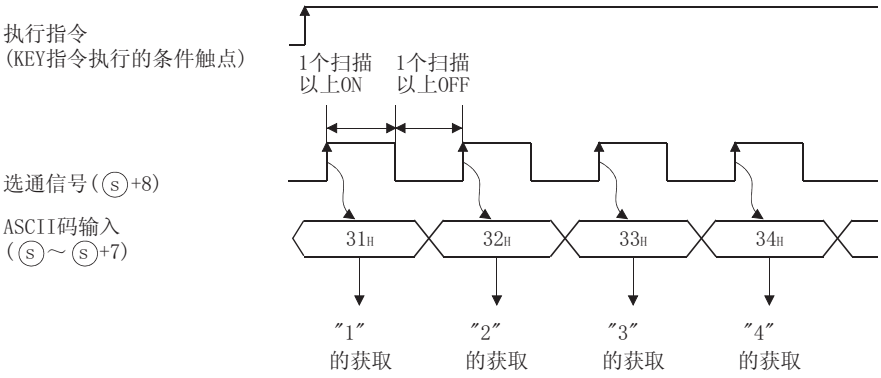
(2) 对ⓢ中指定的输入 (X) 进行数字输入时，将数字对应的 ASCII 码进行位展开后输入到ⓢ～ⓢ+7 中。

可输入的 ASCII 码的范围为 30H(0) ～ 39H(9) 以及 41H(A) ～ 46H(F)。



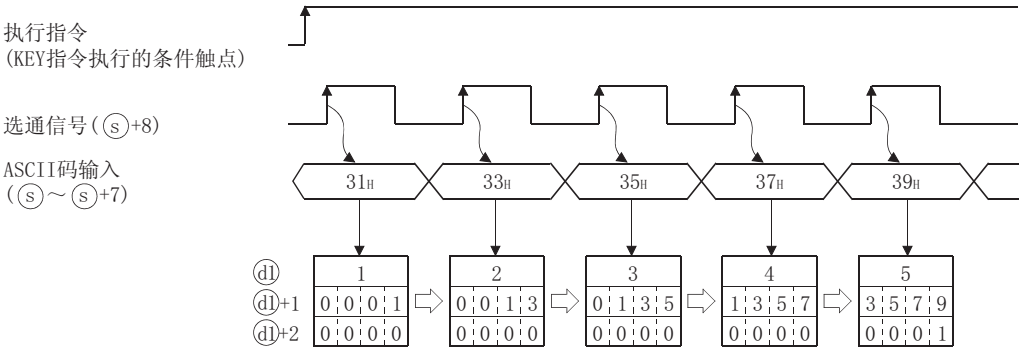
(3) 将 ASCII 码输入到ⓢ～ⓢ+7 中后，通过将ⓢ+8 的选通信号置为 ON，将指定的数字获取到内部。

对于选通信号，应将其 ON/OFF 状态保持顺控程序的 1 个扫描以上。
1 个扫描以下的情况下，有可能无法正常获取数据。



(4) 对于执行指令 (KEY 指令执行的条件触点)，在指定位数的输入结束之前，必须预先置为 ON。执行指令为 OFF 时不能执行 KEY 指令。

(5) 在至ⓓ中指定的软元件的存储中，将实际获取的数字的位数存储到ⓓ中，将输入的 ASCII 码转换为 16 进制 BIN 值后存储到ⓓ+1, ⓓ+2 中。



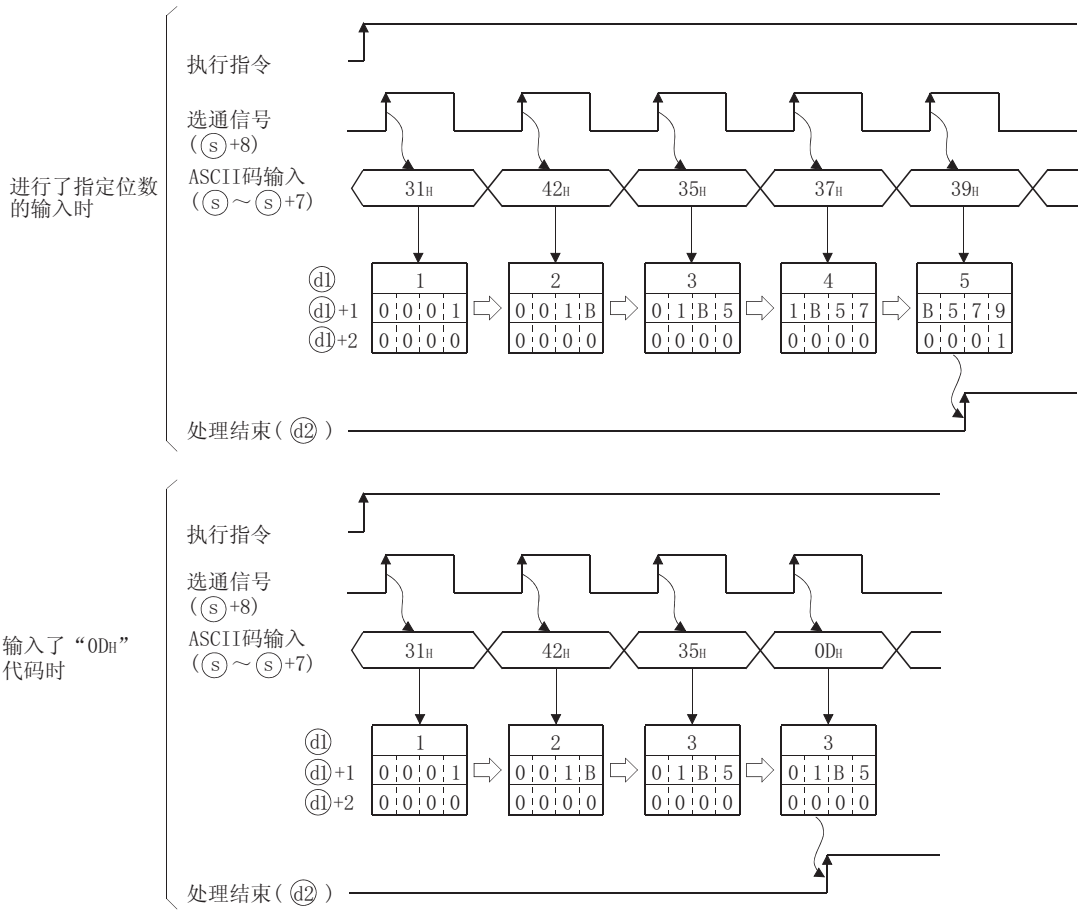
(6) n 中指定的输入位数的范围为 1 ～ 8。

(7) 进行了下述输入的情况下，输入数据至内部的获取将结束，将⑫ 中指定的位软元件置为 ON。

• n 中指定的位数的输入已完毕时

• 输入了 “0DH” 代码时

例如指定了 n=5 时的动作如下述所示。



再次进行输入处理时，需要通过用户程序进行⑪中存储的输入位数、输入数据的清除以及⑫ 中指定的位软元件的 OFF。

如果未进行⑪的清除以及⑫ 的 OFF，将无法进行后续的输入处理。

出 错

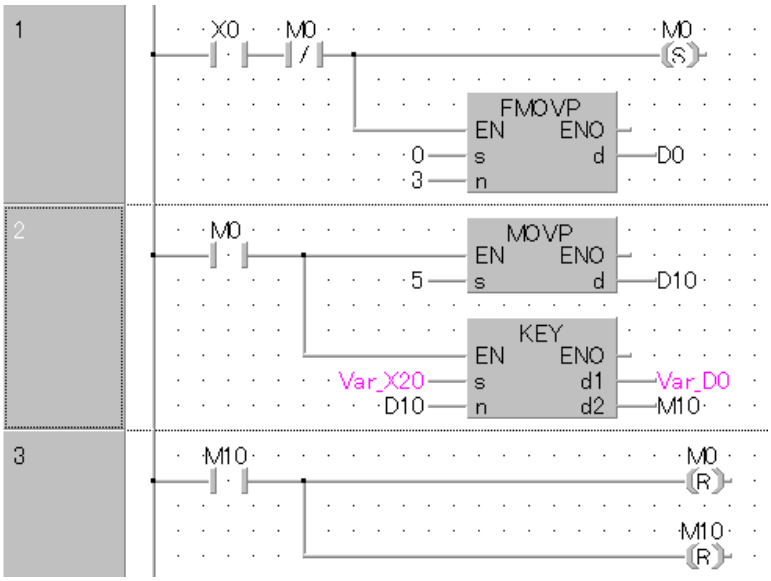
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓓ 中指定的软元件不是输入 (X) 时。 (出错代码 : 4100)
- n 中指定的位数超出了 1 ～ 8 的范围时。 (出错代码 : 4100)

程序示例

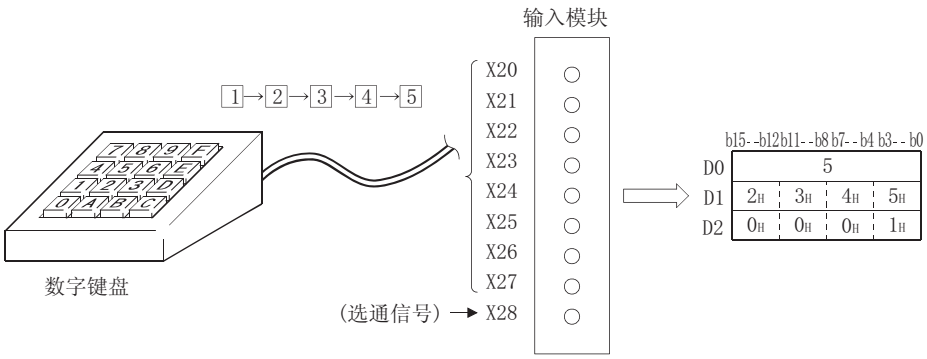
以下为将 X0 置为 ON 时，通过 X20 ～ X28 上连接的数字键盘获取 5 位数以内的数据后，存储到 D0 以后的程序

[结构体梯形图]




```
[ST]
IF X0 AND NOT(M0) THEN
     SET(TRUE,M0);
     FMOVP(TRUE,0,3,D0);
END_IF;
IF M0 THEN
     MOVP(TRUE,5,D10);
     KEY(TRUE,Var_X20,D10,Var_D0,M10);
END_IF;
IF M10 THEN
     RST(TRUE,M0);
     RST(TRUE,M10);
END_IF;
```

[动作]



7.18.8 变址寄存器的批量保存、恢复

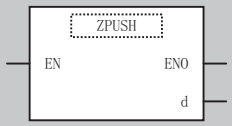
ZPUSH, ZPOP

Basic | High performance | Universal | L CPU

ZPUSH(P)
ZPOP(P)

P: 执行条件 :

结构体梯形图



ST

```
ENO:= ZPUSH (EN, d);
```

[ ] 中放入下述指令。

ZPUSH	ZPUSHP
ZPOP	ZPOPP

输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位
d: 进行变址寄存器的保存 / 恢复的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
ⓓ	-	○		-					

功 能

ZPUSH(P)

(1) 将下述变址寄存器的内容，保存到ⓓ中指定的软元件以后。

（对变址寄存器的内容进行保存时，ⓓ+0（保存次数）将被 +1。）

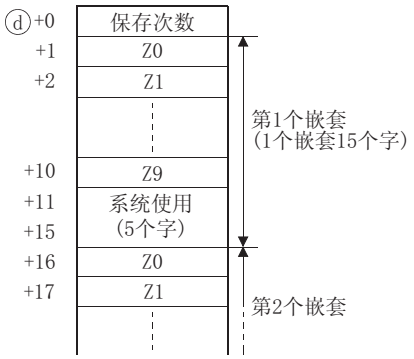
- 基本型 QCPU Z0 ～ Z9
- 高性能型 QCPU Z0 ～ Z15
- 通用型 QCPU、LCPU Z0 ～ Z19

(2) 数据的恢复使用 ZPOP(P) 指令。将 ZPUSH(P) ～ ZPOP(P) 指令成对使用，可以进行嵌套。

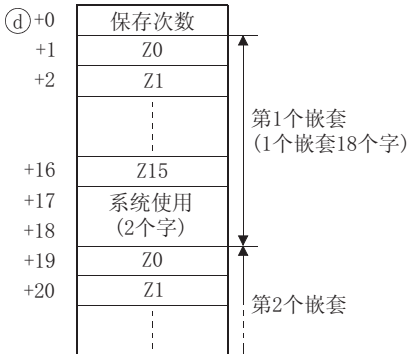
(3) 进行了嵌套的情况下，每次执行 ZPUSH(P) 指令时，使用的区域将被添加到ⓓ以后，因此应预先预留出与使用次数相应的区域。

(4) ⓓ以后使用的区域的构成如下所示。

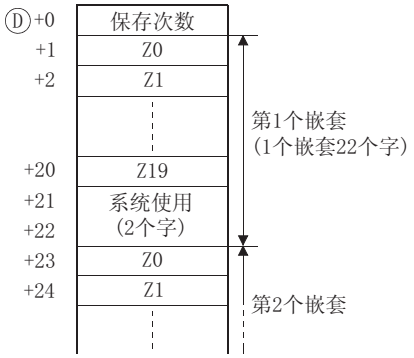
• 使用基本型 QCPU 时



• 使用高性能型 QCPU 时



• 使用通用型 QCPU、LCPU 时



ZPOP(P)

将ⓓ中指定的软元件以后保存的数据，读取到变址寄存器中。（对保存的变址寄存器的内容进行读取时，ⓓ+0（保存次数）将被 -1。）

出 错

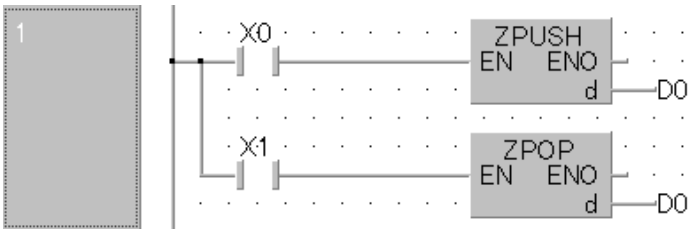
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ZPUSH(P) 指令中ⓓ以后使用的点数的范围超出了相应软元件的范围时。
  （出错代码： 4101）
- ZPOP(P) 指令中ⓓ+0 的内容（保存次数）为 0 时。 （出错代码： 4100）

程序示例

以下为在 P0 以后的子程序内使用变址寄存器时，将子程序调用之前的变址寄存器的内容保存到 D0 以后的程序。

[结构体梯形图]



[ST]

```
ZPUSH(X0,D0);
ZPOP(X1,D0);
```

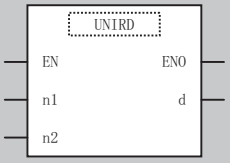
7.18.9 模块信息读取

UNIRD

Basic | High performance | Universal | L CPU

UNIRD(P)	P: 执行条件 :

结构体梯形图



ST

ENO:= UNIRD (EN, n1, n2, d);

[ ] 中放入下述指令。

UNIRD UNIRDP

输入自变量 EN: 执行条件 : 位
n1: 将模块信息读取源起始输入输出编号用 16 相除后的值 (0 ～ FFn) : ANY16
n2: 读取的数据数 (0 ～ 256) : ANY16
输出自变量 , ENO: 执行结果 : 位
d: 存储模块信息的软元件的起始编号 : ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它数据
	位	字		位	字				
n1	○	○		-				○	-
n2	○	○		-				○	-
ⓓ	-	○		-				-	-

功 能

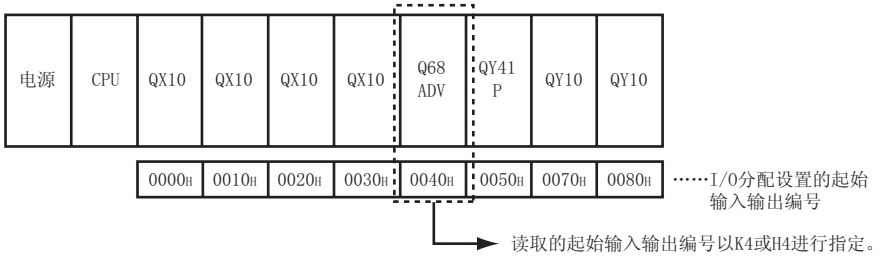
从 n1(将起始输入输出编号用 16 相除后的值)中指定的模块中，将 n2 中指定的点数的模块信息存储到ⓓ中指定的软元件以后。

(不是读取 I/O 分配中指定的模块类型，而是读取实际安装的模块状态。)

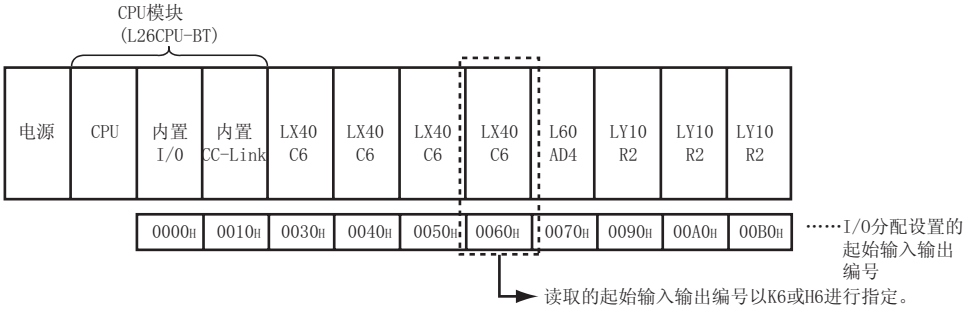
备 注

对于 n1，通过将读取模块信息的插槽的起始输入输出编号以 16 进制数 4 位表示时的高 3 位进行指定。

〈QCPU(Q 模式)〉



〈LCPU〉



模块信息的详细内容如下所示。

位 b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0

各模块信息

位	项目名	内容	
		QCPU(Q 模式)	LCPU
b0	输入输出点数	000:16 点 010:48 点 100:128 点 110:512 点	001:32 点 011:64 点 101:256 点 111:1024 点
b1			
b2			
b3	模块类型	000: 输入模块 001: 输出模块 010: 输入输出混合模块 011: 智能功能模块	000: 输入模块 001: 输出模块 011: 智能功能模块 111:CPU 内置 I/O
b4			
b5			
b6	外部供給电源状态 (将来扩展用)	1: 外部电源供应中 0: 无外部电源供应	0: 固定
b7	保险丝熔断发生有无	1: 有保险丝熔断模块 0: 正常	0: 固定
b8	在线模块更换状态 / 通过待机系统执行	1: 在线模块更换中，或试图从冗余系统的待机系统中读取扩展基板上的模块信息。*1 0: 上述以外	0: 固定
b9	轻・中度出错状态	1: 有轻・中度出错	0: 正常
b10	模块出错状态	00: 无模块出错 10: 轻度出错	01: 中度出错 11: 重度出错
b11			
b12	模块准备状态	1: 正常	0: 有模块出错
b13	空余	0: 固定	
b14	Q 模块	0:Q 系列模块	0: 固定
b15	模块安装状态	1: 模块已安装	0: 模块未安装

*1 ： 对于多 CPU 系统中使用的通用型 QCPU，在其它机号中管理的模块处于在线模块更换中的情况下也将变为 ON。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

[高性能型 QCPU、通用型 QCPU、L26CPU-BT]

- n1 超出了 0 ～ FFH 的范围时。 (出错代码： 4100)
- n2 超出了 0 ～ 256 的范围时。 (出错代码： 4100)
- n1 与 n2 的合计为 257 或以上时。 (出错代码： 4100)

[Q00/Q01CPU, L02CPU]

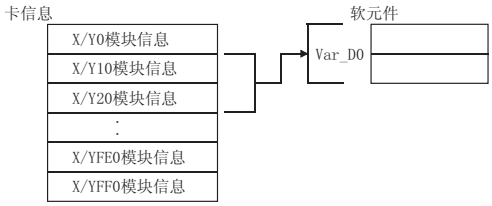
- n1 超出了 0 ～ 3FH 的范围时。 (出错代码： 4100)
- n2 超出了 0 ～ 64 的范围时。 (出错代码： 4100)
- n1 与 n2 的合计为 65 或以上时。 (出错代码： 4100)

[Q00JCPU]

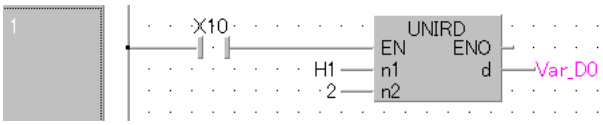
- n1 超出了 0 ～ FH 的范围时。 (出错代码： 4100)
- n2 超出了 0 ～ 16 的范围时。 (出错代码： 4100)
- n1 与 n2 的合计为 17 或以上时。 (出错代码： 4100)

程序示例

以下为将 X10 置为 ON 时，将输入输出编号为 10H ～ 20H 的模块信息存储到 Var_D0 以后的程序。



[结构体梯形图]

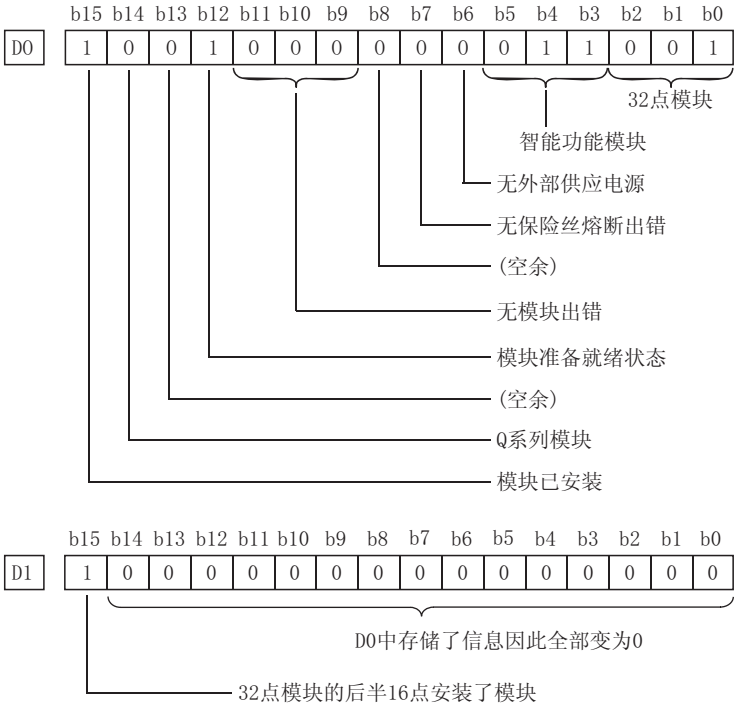


[ST]

```
UNIRD(X10,H1,2,Var_D0);
```

读取结果侧（读取至 D0 时）

1）Q 系列 32 点智能功能模块的情况



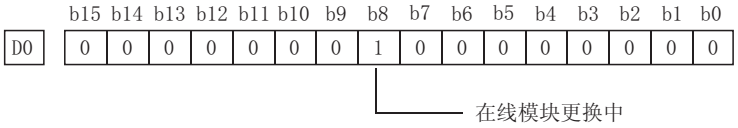
- 48 点模块时在 D2 中存储与 D1 相同的内容，64 点模块时在 D2 与 D3 中分别存储与 D1 相同的内容。

2）A 系列 32 点模块

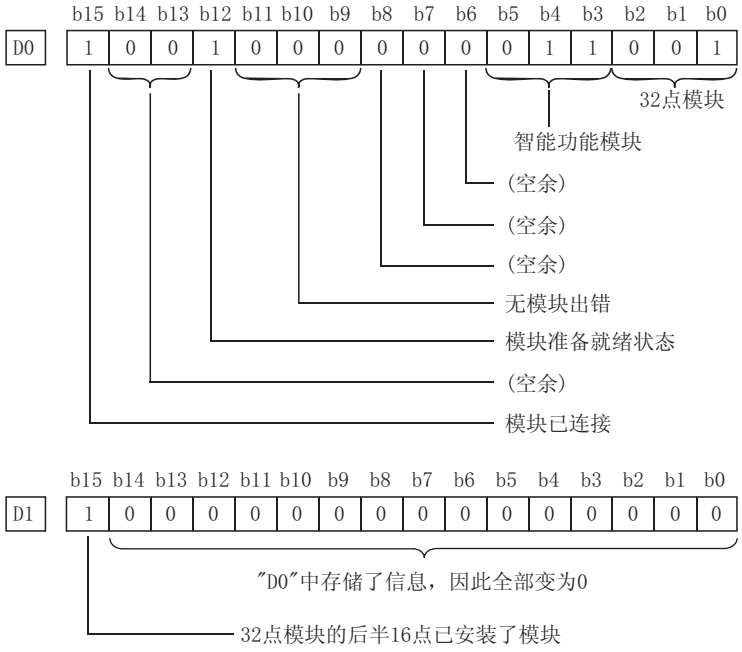
- 48 点模块时在 D2 中存储与 D1 相同的内容，64 点模块时在 D2 与 D3 中分别存储与 D1 相同的内容。

3）空余插槽

4) 在线模块更换中



5) LCPU 32 点智能功能模块的情况



7.18.10 模块型号读取

TYPERD

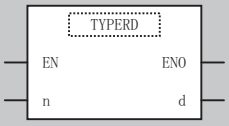
Basic | High performance | Ver. Universal | L CPU

通用型 QCPU：序列号的前 5 位数为“11043”以后

TYPERD(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= TYPERD (EN, n, d);

□ 中放入下述指令。

TYPERD　　TYPERDP

输入自变量　EN:　执行条件　：位

n:　将读取模块型号的模块的起始输入输出编号用 16 相除后的值　：ANY16

输出自变量　ENO:　执行结果　：位

d:　指令执行结果保存的起始软元件　：ANY16(0..9)

- d[0]：指令执行结果
- d[1] ～ d[9]：模块型号

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它数据
	位	字		位	字				
n	｜	○		－				○	－
ⓓ	－	○		－				－	－

设置数据/控制数据

设置数据	内容		设置范围	设置侧	数据类型
n	将读取模块型号的模块的起始输入输出编号用 16 相除后的值		0 ～ FFH, 3E0 ～ 3E3H*1	用户	BIN16 位
ⓓ	ⓓ[0]	指令执行结果	各软元件的范围内	系统	BIN16 位
	ⓓ[1] ～ ⓓ[9]	模块型号			字符串

*1 ：　仅通用型 QCPU

功 能

(1) 对 n 中指定的插槽的模块型号进行读取后，存储到ⓓ中指定的软元件以后。

通用型 QCPU 的情况下，对象模块为下述 6 种。

- CPU 模块
- 输入模块
- 输出模块
- 输入输出混合模块
- 智能功能模块
- GOT(总线连接时)

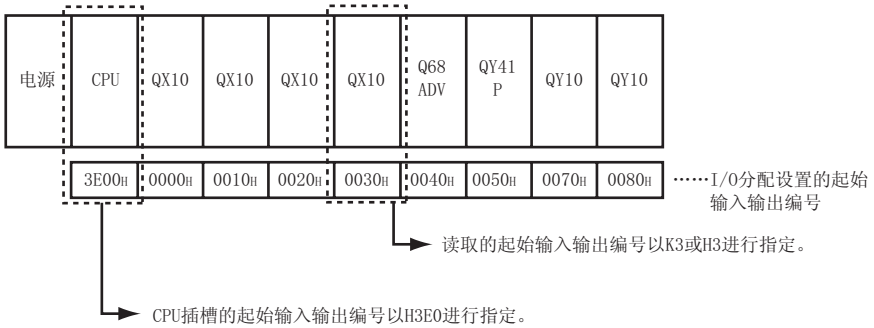
LCPU 的情况下，对象模块为下述 4 种。

- CPU 模块
- 输入模块
- 输出模块
- 智能功能模块

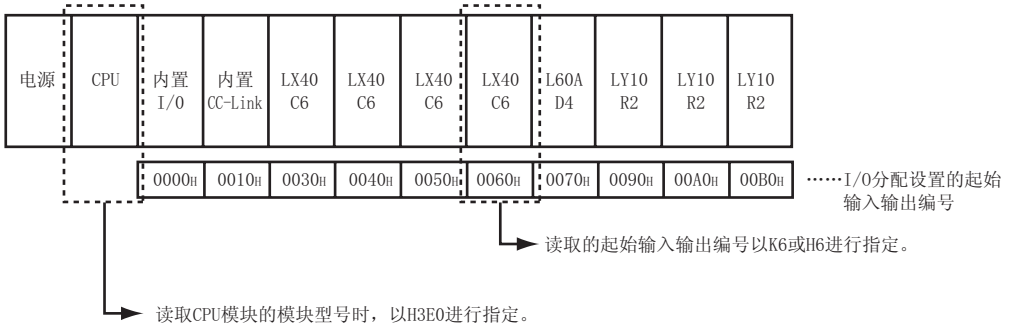
(2) 在 n 中按下述方式对读取对象模块的起始输入输出编号进行指定。

- 指定将读取对象模块的起始输入输出编号用 16 相除后的值。

〈通用型 QCPU 〉



〈LCPU〉



要 点

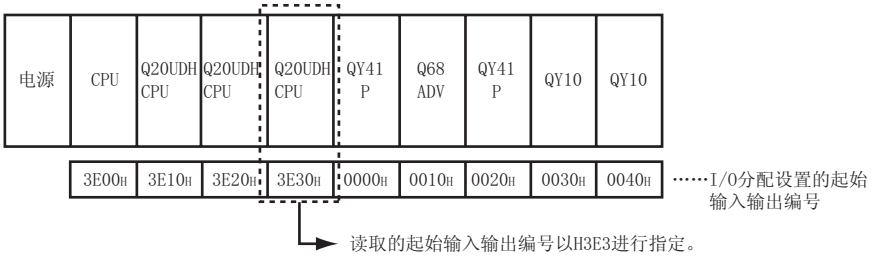
在 LCPU 中，指定了内置 I/O、内置 CC-Link 的起始 I/O 的情况下，CPU 模块的型号将被读取。

• 指定占用 2 个插槽的模块的情况
读取对象模块中指定的起始输入输出编号有时与安装插槽的起始输入输出编号不相同。
关于指定的起始输入输出编号请参阅各模块手册。
指定将读取对象模块的起始输入输出编号用 16 相除后的值。

例 QJ71GP21S-SX 的情况下

指定的起始输入输出编号为与已安装模块的 0010H 相加后的值。

• 多 CPU 系统配置时，读取 CPU 模块型号的情况
指定将各 CPU 机号的起始输入输出编号用 16 相除后的值。

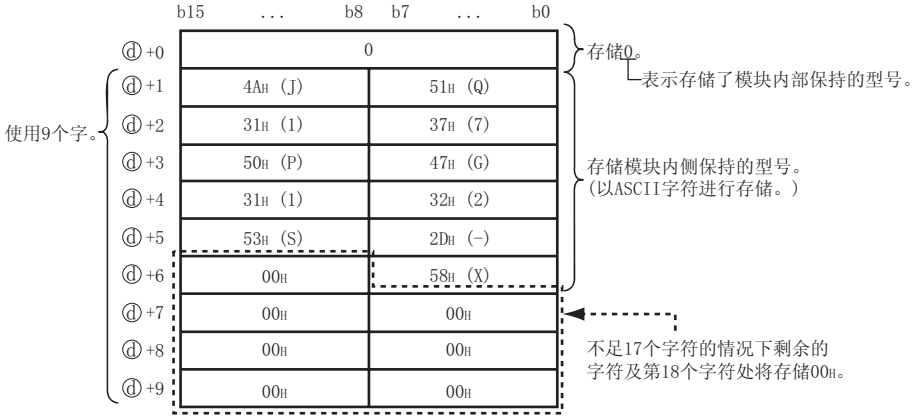


此外，即使指定了其它机号 CPU 管理的模块的起始输入输出编号也可读取模块型号。

(3) ⓓ[0]中存储指令执行结果，ⓓ[1]～ⓓ[9]中存储模块型号。

ⓓ中存储的值如下所示。

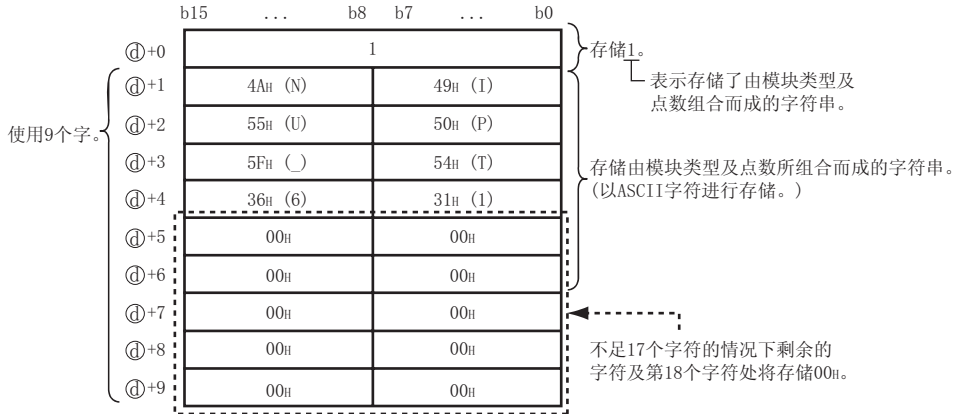
(a) 读取对象模块内部保持有型号的情况下（例： QJ71GP21-SX)



ⓓ[1]～ⓓ[9]中存储的型号示例如下所示。

对象模块	存储的型号示例
CPU 模块	Q06UDEHCPU
智能功能模块	QJ71GP21-SX
GOT	GOT1000

(b) 读取对象模块未保持有内部型号的情况下（例： QX40)



ⓓ[1]～ⓓ[9]中存储的字符串例如下所示。

对象模块	存储的型号示例
输入模块	INPUT_16
输出模块	OUTPUT_32
输入输出混合模块	MIXED_64
智能功能模块	INTELLIGENT_128

[表示模块类型的字符串]

- 输入模块： INPUT
- 输出模块： OUTPUT
- 输入输出混合模块： MIXED
- 智能功能模块*1: INTELLIGENT

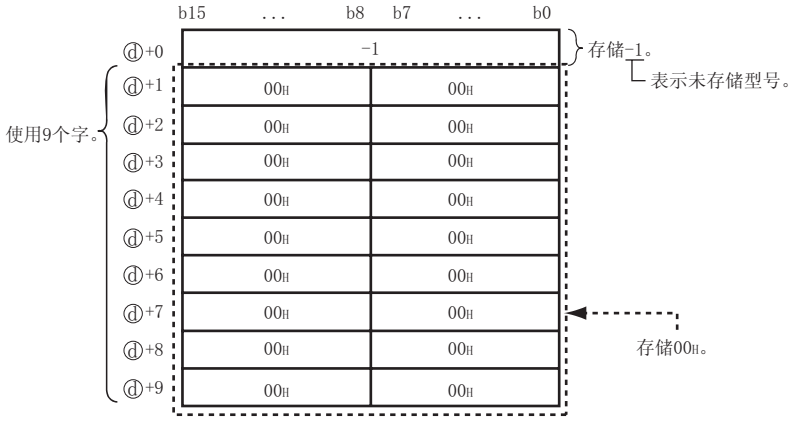
*1: 包括 QI60、GOT。

[表示模块类型的字符串]

- 16 点： _16
- 32 点： _32
- 64 点： _64
- 128 点： _128
- 256 点： _256
- 512 点： _512
- 1024 点： _1024

(c) 其它

- 空余插槽或在线模块更换中的情况下
- n 不是模块的起始输入输出编号的情况下
- n 的指定在设置范围内，但不是可编程控制器参数的 I/O 分配中可设置的值的情况下



出 错

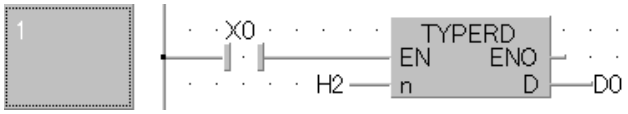
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 由于读取对象模块故障等原因导致不能通信时。 (出错代码： 2110)
- ⓓ 中指定的软元件算起的 10 个字超出了可使用的软元件范围时。 (出错代码： 4101)
- n 的指定超出了 0 ～ FFH、3E0 ～ 3E3H 的范围时。(通用型 QCPU 时) (出错代码： 4101)
- n 的指定超出了 0 ～ FFH 的范围时。(LCPU 时) (出错代码： 4101)

程序示例

以下为 X0 变为 ON 时，将起始输入输出编号 0020H 中安装的模块的模块型号存储到 D0 以后的程序。

[结构体梯形图]



[ST]

```
TYPERD(X0,H2,D0);
```

7.18.11 跟踪设置、复位

TRACE, TRACER

Basic | High performance | Universal | L CPU

TRACE
TRACER

结构体梯形图

ST

```
ENO:= TRACE (EN);
```

□ 中放入下述指令。

TRACE
TRACER

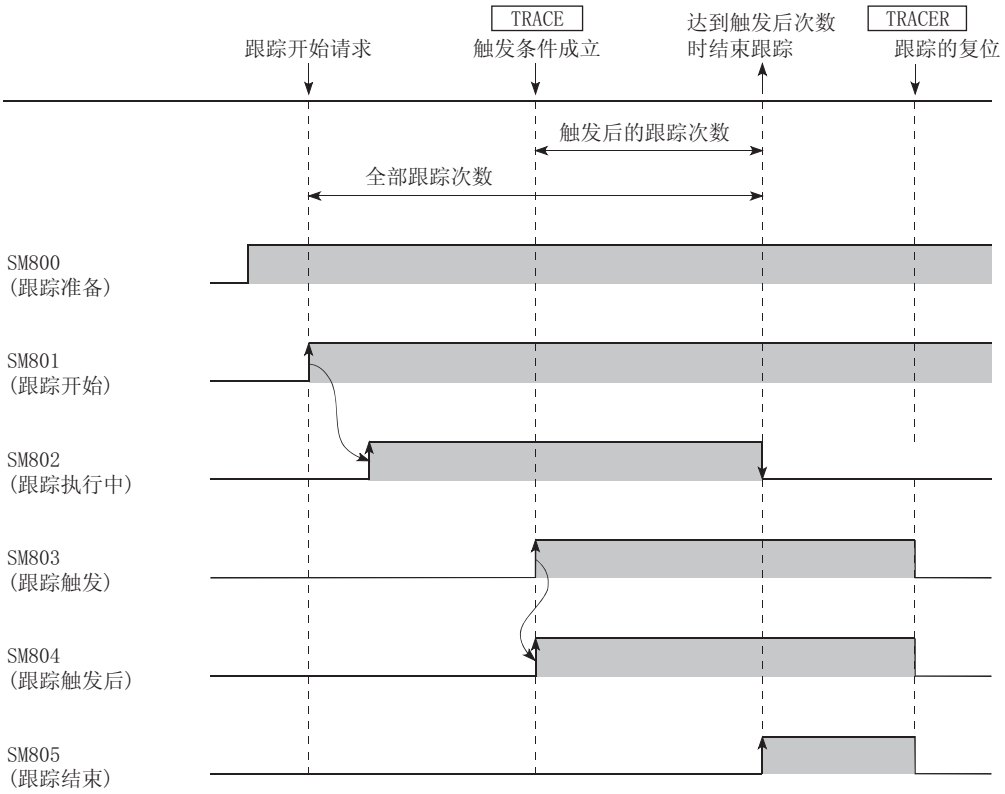
输入自变量， EN: 执行条件 : 位
输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数	其它数据
	位	字		位	字				
–	–								

功 能

跟踪功能是指，在指定的时机对 CPU 模块的指定软元件的内容进行连续采集的功能。

进行跟踪的情况下，在 SM800、SM801、SM802 变为 ON 时，将所设置的次数的跟踪结果存储到跟踪用文件中。



TRACE

(1) 执行 TRACE 指令时，将 SM803 置为 ON，按照跟踪条件设置的触发后次数中设置的次数进行采样后，对数据进行锁存并停止采样跟踪。

(2) 跟踪执行过程中如果 SM801 变为 OFF，则停止采样。

(3) 执行 TRACE 指令后，跟踪结束，SM805 变为 ON。

(4) 如果执行了 1 次 TRACE 指令，则从第 2 次起将被忽略。
如果执行了 TRACER 指令，则 TRACE 指令将重新变为有效。

TRACER

(1) TRACER 指令是对 TRACE 指令进行复位的指令。
如果执行了 TRACER 指令，则 TRACE 指令将重新变为有效。

(2) 如果执行了 TRACER 指令，SM803 ～ SM805 将变为 OFF。

备 注

1. 关于跟踪的详细内容，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。
2. 通过 GX Works2 执行跟踪时，请参阅编程工具的操作手册。

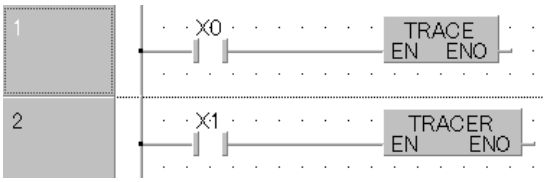
出 错

不存在 TRACE、TRACER 指令相关的运算出错。

程序示例

以下为 X0 变为 ON 时执行 TRACE 指令，X1 变为 ON 时通过 TRACER 指令对 TRACE 指令进行复位的程序。

[结构体梯形图]




```
[ST]
TRACE(X0);
TRACER(X1);
```

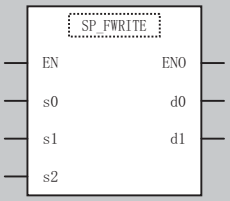
7.18.12 至指定文件的数据写入

SP_FWRITE

Basic | High performance | Universal | L CPU

SP_FWRITE

结构体梯形图



ST

ENO:= SP_FWRITE (EN, s0, s1, s2, d0, d1);

[ ] 中放入下述指令。

SP_FWRITE

输入自变量， EN: 执行条件 : 位
s0: 驱动器的指定 : ANY16
s1: 文件名 : 字符串
s2: 写入请求的数据数 : ANY16
输出自变量， ENO: 执行结果 : 位
d0: 控制数据 : ANY16 的数组 (0..7)
d1: 通过处理结束置为 ON 的数组 : 位的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它数据
	位	字		位	字			K, H	$	
ⓢ0	○	○		-				○	-	-
ⓢ1	-	○		-				-	○	-
ⓢ2	-	○		-				○	-	-
ⓓ0	-	○		-				-	-	-
ⓓ1	△*1	△*1		-				-	-	-

*1 ：不能使用局部软元件以及各程序中设置的文件寄存器。

设置数据/控制数据

设置数据	内容			设置范围	设置侧	数据类型
U0	虚拟			–	–	BIN16 位
ⓢ0	驱动器指定			2	用户	
ⓢ1	存储文件名的软元件的起始编号。文件名如下所示。					
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓢ1 ～ ⓢ1 +□	文件名字符串	指定文件名的字符串。 • 省略扩展名的情况下应从 “.”（点号）开始省略。 • 应在 8 个字符 + 点号 +3 个字符以内进行指定。 • 指定了 9 个字符或以上时，即使有扩展名也将被忽略，扩展名将变为 “BIN”或者 “CSV”。	字符串	用户	
ⓢ2	存储数据的软元件的起始编号。写入数据如下所示。					
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓢ2	请求写入数据数	指定写入请求的数据数。（字单位） ⓓ10 +7 中指定字节时也进行字换算以字单位进行设置。	1 ～ 480 1 ～ 32707*2	用户	
	ⓢ2 +1 ～ ⓢ2 +□	写入数据	进行写入请求的数据	0000_H ～ $FFFF_H$		

*2: 为通用型 QCPU、LCPU 的范围。

<table>
<tr><th>设置数据</th><th colspan="3">内容</th><th>设置范围</th><th>设置侧</th><th>数据类型</th></tr>
<tr><td rowspan="10">(d0)</td><td colspan="5">存储控制数据的软元件的起始编号。
控制数据如下所示。</td><td rowspan="10">BIN16 位</td></tr>
<tr><td>软元件</td><td>项目</td><td>内容以及设置数据</td><td>设置范围</td><td>设置侧</td></tr>
<tr><td>(d0) [0]</td><td>执行 / 结束类型</td><td>指定执行类型。
0000H : 二进制写入
0100H : CSV 格式转换写入</td><td>0000H
0100H</td><td>用户</td></tr>
<tr><td>(d0) [1]</td><td>（未使用）</td><td>系统使用</td><td>-</td><td>系统</td></tr>
<tr><td>(d0) [2]</td><td>写入结果（数据数）</td><td>存储对 (s2) 中指定的数据实际进行写入的数据数。值的单位取决于数据类型指定。</td><td>-</td><td>系统</td></tr>
<tr><td>(d0) [3]</td><td>（未使用）</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>(d0) [4]
(d0) [5]</td><td>文件位置</td><td>(d0) 中指定二进制写入时
• 指定文件位置。
00000000H: 从文件的起始开始
00000001H ～ FFFFFFFEH
: 根据指定位置
单位取决于数据类型指定。
FFFFFFFFH : 添加在文件的最后
(d0) 中指定 CSV 格式写入时
• 对于序列号的前 5 位数为 “01111” 以前的高性能型 QCPU，必须设置为文件的起始 (0H)。
• 对于序列号的前 5 位数为 “01112” 以后的高性能型 QCPU/ 通用型 QCPU/LCPU，设置文件位置。
00000000H ～ FFFFFFFEH
: 从文件的起始开始
FFFFFFFFH : 添加到文件的最后</td><td>00000000H ～ FFFFFFFFH</td><td>用户</td></tr>
<tr><td>(d0) [6]</td><td>列数指定</td><td>(d0) 中指定为二进制写入时必须设置为 0。
(d0) 中指定为 CSV 格式写入时，设置进行写入的列数。
0 : 无列。设置为 1 行。
0 以外 : 指定数的列。</td><td>0 ～ 65535</td><td>用户</td></tr>
<tr><td>(d0) [7]</td><td>数据类型指定</td><td>0: 字
1: 字节</td><td>0,1</td><td>用户</td></tr>
<tr><td rowspan="4">(d1)</td><td colspan="5">通过处理结束置为 ON 的位软元件（但是异常结束时 (d1)+1 也置为 ON。）</td><td rowspan="4">位</td></tr>
<tr><td>软元件</td><td>项目</td><td>内容以及设置数据</td><td>设置范围</td><td>设置侧</td></tr>
<tr><td>(d1) [0]</td><td>结束信号</td><td>表示处理结束。
ON: 结束 OFF: 未结束</td><td>-</td><td rowspan="2">系统</td></tr>
<tr><td>(d1) [1]</td><td>异常结束信号</td><td>表示是正常结束还是异常结束。
ON: 异常结束 OFF: 正常结束</td><td>-</td></tr>
</table>

注意事项

(1) QCPU(Q 模式）的情况下，(s0)（驱动器指定）中只能指定 ATA 卡的驱动器 (2)。
但是安装了 Flash 卡的情况下，不能通过 SP_FWRITE 指令进行写入。
不能设置为 SRAM 卡、标准 RAM、标准 ROM 的驱动器。
LCPU 的情况下，(s0)（驱动器指定）中只能设置 SD 存储卡的驱动器 (2)。

(2) CSV 设置时写入的数据为 10 进制数的值。

例 字符 “A”(41H) → 65 被写入。

可用范围 : -32768 ～ 32767

(3) 二进制写入时，字指定中的文件位置的设置范围为 00000000H ～ 7FFFFFFFH, FFFFFFFFH。

功 能

(1) 将指定的数据数的数据写入到指定的文件中。

根据控制数据的执行 / 结束类型，指定是直接以二进制数据写入，还是将二进制数据转换为 CSV 格式后进行写入。

(QCPU(Q 模式) 的情况下写入的对象只能为 ATA 卡，LCPU 的情况下写入的对象只能为 SD 存储卡。)

(2) 对于处理结束ⓓ1的位软元件，在检测到本指令的处理结束的 END 执行时将自动地置为 ON，通过下一个扫描的 END 置为 OFF。

作为本指令的执行结束标志使用。

本指令异常结束时，异常结束ⓓ1[1] 的软元件与处理结束ⓓ1[0] 的软元件在相同的时机 ON/OFF，因此作为本指令的异常结束标志使用。

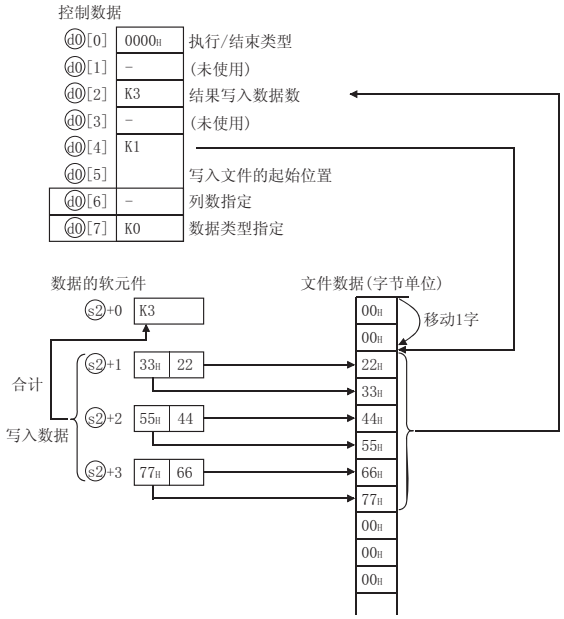
此外，指令执行过程中 SM721 将为 ON。

SM721 处于 ON 的状态下，不能执行本指令。(执行的情况下将成为无处理。)

再者，在执行指令时检测出的出错 (SM721 为 ON 之前) 中，处理结束 [0]、异常结束ⓓ1[1] 以及 SM721 不变为 ON。

(3) 数据的请求写入数据数 (ⓢ2) 以及文件位置ⓓ0[4], ⓓ0[5] 的处理单位应设置为字单位。

二进制写入的情况下的请求写入数据数、指定文件位置的情况下的数据的写入方法如下所示。



(4) 二进制写入时

(a) 省略了对象文件的扩展名的情况下，扩展名将变为 “.BIN”。

(b) 指定了不存在的文件的情况下，新建相应文件，从起始开始对数据进行添加保存。
此时新建文件的属性被设置为存档。

(c) 在数据的写入过程中，超过了已有的容量的情况下，超出部分的数据将被添加保存。

(d) 指定了大于已有文件容量的文件位置时的情况如下所示。
- 对于序列号的前 5 位数为 “01111”以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112”以后的高性能型 QCPU / 通用型 QCPU，将变为写入 0 点并正常结束。

(e) 在对数据进行添加保存的过程中，如果存储媒介没有空余区域的情况下将变为出错状态。
此时，已成功写入以及添加保存的部分将保持为被写入状态不变。
在可添加保存部分被添加保存后出错结束。

(5) CSV 格式转换写入时

(a) 在省略了扩展名的情况下，扩展名将变为 “.CSV”。

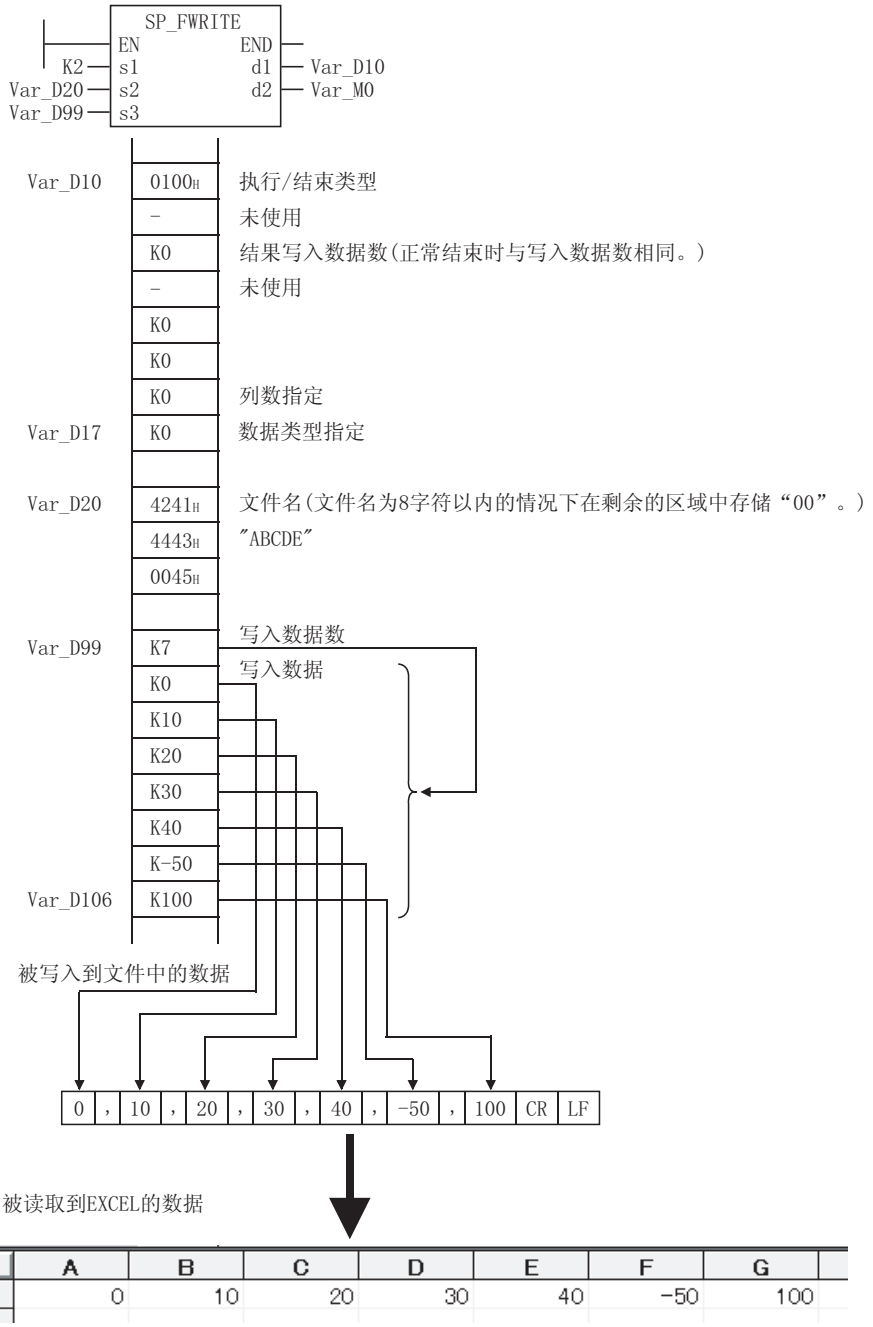
(b) 指定了已存在的文件时的情况如下所示。
[序列号的前 5 位数为 “01111”以前的高性能型 QCPU]
将文件内容全部删除后，从起始开始保存数据。
[序列号的前 5 位数为 “01112”以后的高性能型 QCPU / 通用型 QCPU]
- ⑩[4],⑩[5] 中设置了除 FFFFFFFFH 以外时，将文件内容全部删除后，从起始开始保存数据。
- ⑩[4],⑩[5] 中设置为 FFFFFFFFH 时，从文件的最后开始保存数据。

(c) 指定了不存在的文件的情况下，新建相应文件后，从起始开始添加保存数据。
此时新建文件的属性被设置为存档。

(d) 在对数据进行添加保存的过程中，如果存储媒介没有空余区域的情况下将变为出错状态。
此时，已成功写入以及添加保存的部分将保持为被写入状态不变。
在可添加保存部分被添加保存后出错结束。

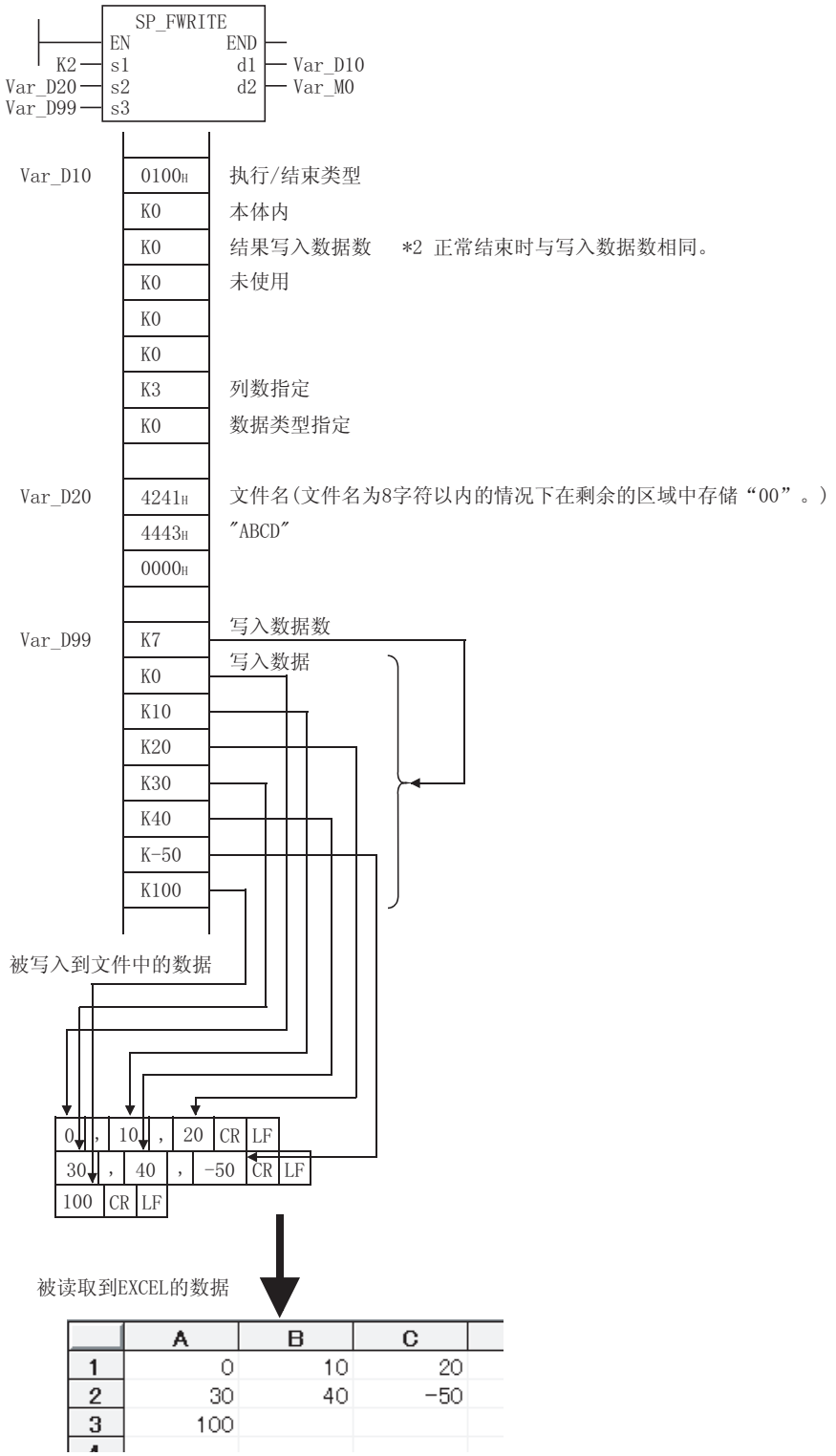
(e) 指定列数为 0 的情况下，保存为 1 行的 CSV 格式文件。

例 CSV 格式转换写入时指定列数为 0 的情况下



(f) 在 CSV 格式转换写入中，指定列数为 0 以外的情况下，将被保存为指定列数的表格的 CSV 格式文件。

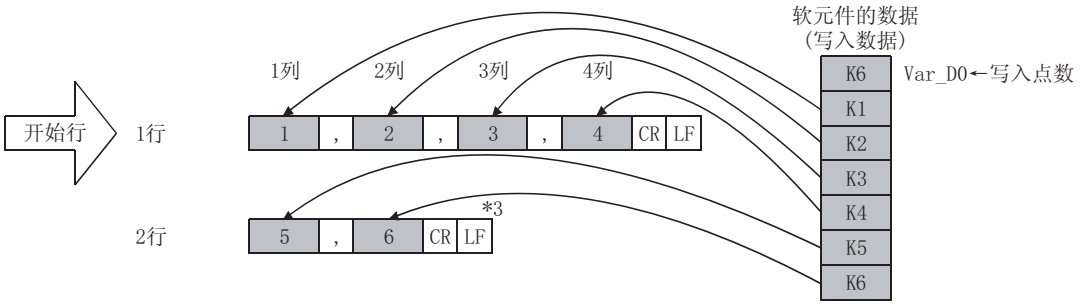
例 CSV 格式转换写入时指定列数为 0 以外的情况下



(g) 在序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 通用型 QCPU/LCPU 中，进行数据添加时的情况如下所示。

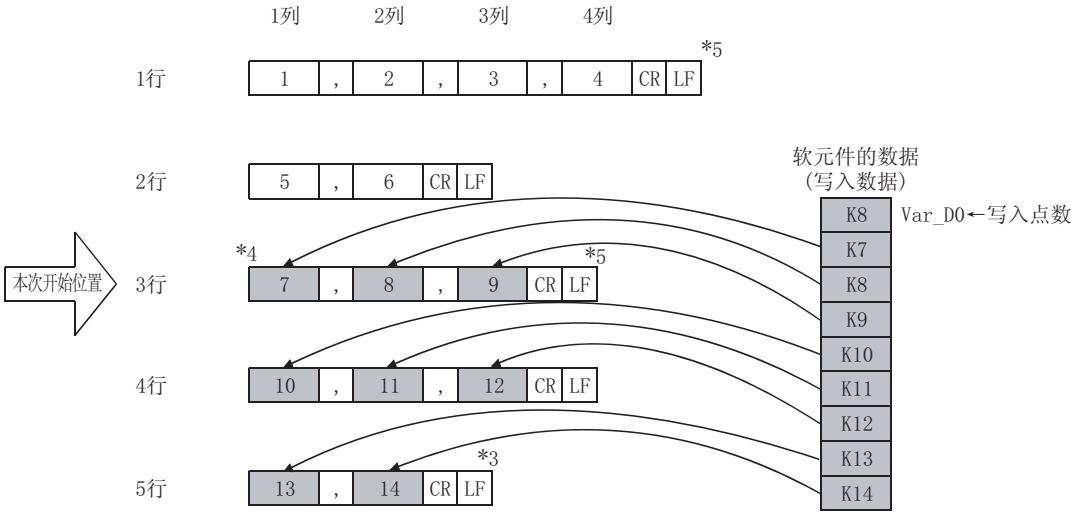
[指定进行写入的文件。](即使文件已存在也将其删除后重新创建。)

执行类型	= CSV格式	文件位置	= 0H(重新创建文件)
列指定	= 4H*3*5	写入起始软元件	= Var_D0
数据类型指定	= 字	数据数	= 6H*3



[通过添加模式添加到文件的最后。]

执行类型	= CSV格式	文件位置	= FFFFFFFFH(继续模式)
列指定	= 3H*3*5	写入起始软元件	= Var_D7
数据类型指定	= 字	数据数	= 8H*3



*3: “写入点数”不是 “列指定”的整数倍时列数将变为错乱。
*4: 由于最后的数据的后面必须输入换行码，因此通常添加模式时从新的一行的起始处开始添加。
*5: 添加模式时，“列指定”与上次写入时不相同的情况下列数将错乱。

(h) 在中断程序中不要执行本指令。
(在中断程序中执行时将无法保证动作正常。)

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

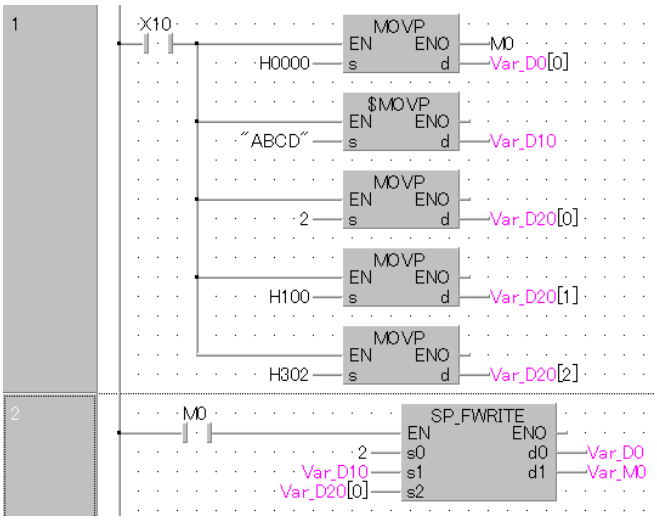
- 驱动器指定(s0)中指定的驱动器为 ATA 卡以外时。
  (QCPU(Q 模式) 时) (出错代码 : 4100)
- 驱动器指定(s0)中指定的驱动器为 SD 存储卡以外时。(LCPU 时)
  (出错代码 : 4100)
- 控制数据(d0)以后中设置的值超出了设置范围时。 (出错代码 : 4100)
- 请求写入数据数(s2)中指定的值超出了设置范围时，或超出了(s2)+1 以后的软元件范围时。
  (出错代码 : 4101)
- ATA 卡的空余容量不足时。(QCPU(Q 模式) 时) (出错代码 : 4100)
- SD 存储卡的空余容量不足时。(LCPU 时) (出错代码 : 4100)
- 试图新建文件的情况下没有空余容量时。 (出错代码 : 4100)
- 指定了不能指定的软元件时。 (出错代码 : 4004)
- ATA 卡内发生了访问异常时。(QCPU(Q 模式) 时) (出错代码 : 4100)
- SD 存储卡内发生了访问异常时。(LCPU 时) (出错代码 : 4100)
- 文件名 ((s1)) 中设置了不能使用的值时。 (出错代码 : 4100)
- 文件名 ((s1)) 的属性为只读时。 (出错代码 : 4100)
- (d0)或(d1)中指定的软元件超出了相应软元件的范围时。(通用型 QCPU、LCPU 时)
  (出错代码 : 4101)

程序示例

(1) 以下为将 X10 置为 ON 时，在驱动器 2 中安装的存储卡的文件 “ABCD.BIN”中添加 00H、01H、02H、03H 的 4 个字节的二进制数据的程序。

- 对于控制数据用软元件，假设已预留了 ⓓ⓪ 算起的 8 点。

[结构体梯形图]

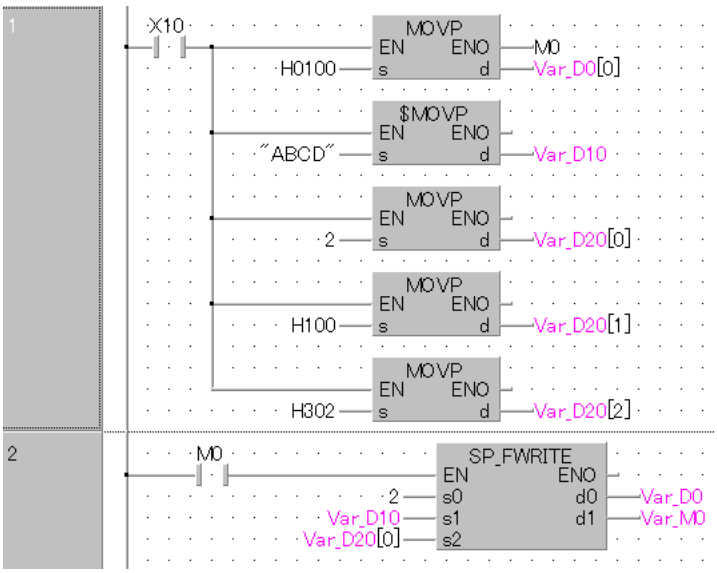



```
[ST]
IF X10 THEN
     MOVP(TRUE,H0000,Var_D0[0]);
     Var_D10:="ABCD";
     MOVP(TRUE,2,Var_D20[0]);
     MOVP(TRUE,H100,Var_D20[1]);
     MOVP(TRUE,H302,Var_D20[2]);
     SP_FWRITE(TRUE,2,Var_D10,Var_D20[0],Var_D0,Var_M0);
END_IF;
```

(2) 以下为将 X10 置为 ON 时，将 00H、01H、02H、03H 的 4 个字节以 2 列的 CSV 格式文件使用文件名 “ABCD.CSV”创建到安装在驱动器 2 中的存储卡中的程序。

- 被写入的文件如下所示。

[结构体梯形图]



```
[ST]
IF X10 THEN
     MOVP(TRUE,H0100,Var_D0[0]);
     Var_D10:="ABCD";
     MOVP(TRUE,2,Var_D20[0]);
     MOVP(TRUE,H100,Var_D20[1]);
     MOVP(TRUE,H302,Var_D20[2]);
     SP_FWRITE(TRUE,2,Var_D10,Var_D20[0],Var_D0,Var_M0);
END_IF;
```

• 对于控制数据用软元件，假设已预留了⑩算起的 8 点。

0	,	1	,	CR	LF	被写入的
2	,	3	,	CR	LF	文件内容

被读取到EXCEL中的数据

	A	B
1	0	1
2	2	3

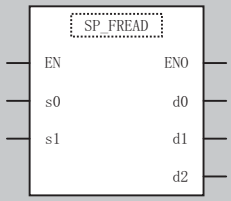
7.18.13 从指定文件的数据读取

SP_FREAD

Basic | High performance | Universal | L CPU

SP_FREAD

结构体梯形图



ST

```
ENO:= SP_FREAD (EN, s0, s1, d0, d1, d2);
```

[ ] 中放入下述指令。

SP_FREAD

输入自变量， EN: 执行条件 : 位
s0: 驱动器的指定 : ANY16
s1: 文件名 : 字符串
输出自变量， ENO: 执行结果 : 位
d0: 控制数据 : ANY16 的数组 (0..7)
d1: 存储读取的数据的软元件的起始编号 : ANY16
d2: 通过处理结束置为 ON 的数组 : 位的数组 (0..1)

设置数据	内部软元件		R, RZ	J□\□		U□\G□	Zn	常数		其它数据
	位	字		位	字			K, H	$	
ⓢ0	○	○		-				○	-	-
ⓢ1	-	○		-				-	○	-
ⓓ0	-	○		-				-	-	-
ⓓ1	-	○		-				-	-	-
ⓓ2	△*1	△*1		-				-	-	-

*1 ：不能使用局部软元件以及各程序中设置的文件寄存器。

设置数据/控制数据

设置数据	内容			设置范围	设置侧	数据类型
U0	虚拟			-	-	
ⓢ0	驱动器指定			2	用户	
ⓢ1	存储文件名的软元件的起始编号。文件名如下所示。					
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓢ1 ～ ⓢ1 +□	文件名字符串	指定文件名的字符串。 • 省略扩展名的情况下应从 “.”（点号）开始省略。 • 应在 8 个字符 + 点号 +3 个字符以内进行指定。 • 指定了 9 个字符或以上时，即使有扩展名也将被忽略，扩展名将变为 “BIN”或者 “CSV”。	字符串	用户	
ⓓ0	存储控制数据的软元件的起始编号。 控制数据如下所示。					
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓓ0 [0]	执行 / 结束类型	指定执行类型。 0000_H ： 二进制读取 0100_H ： CSV 格式转换读取	0000_H 0100_H	用户	
	ⓓ0 [1]	（未使用）	系统使用	-	系统	
	ⓓ0 [2]	请求读取的数据数	指定希望读取的数据数。 （字单位） ⓓ0 [7] 中指定了字节时也进行字换算后以字单位进行设置。	1 ～ 480 1 ～ 32767*2	用户	BIN16 位
	ⓓ0 [3]	（未使用）	-	-	-	
	ⓓ0 [4] ⓓ0 [5]	文件位置	ⓓ0 中指定二进制读取时 • 指定文件位置。 00000000_H: 从文件的起始开始 00000001_H ～ $FFFFFFFE_H$ ：根据指定位置 单位取决于数据类型指定。 $FFFFFFFF_H$ ：不能设置 ⓓ0 中指定 CSV 格式读取时 • 对于序列号的前 5 位数为 “01111”以前的高性能型 QCPU，必须设置为文件的起始 (0_H)。 • 对于序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 通用型 QCPU/LCPU，设置文件位置。 00000000_H: 从文件的起始开始读取 00000001_H ～ $FFFFFFFE_H$ ：从指定行开始读取 $FFFFFFFF_H$ ：从上次的读取位置开始继续读取	00000000_H ～ $FFFFFFFE_H$, $FFFFFFFF_H$	用户	

*2 ： 为通用型 QCPU、LCPU 的范围。

设置数据	内容			设置范围	设置侧	数据类型
ⓓ⓪	ⓓ⓪ [6]	列数指定	ⓓ⓪ 中指定为二进制读取时必须设置为 0。 ⓓ⓪ 中指定为 CSV 格式读取时，设置进行读取的列数。 0 ：无列。视为 1 行。 0 以外 ：视为指定数的列。	0, 1 ～ 65535	用户	BIN16 位
	ⓓ⓪ [7]	数据类型指定	0：字 1：字节	0, 1	用户	
ⓓ①	存储读取的数据的软元件的起始编号。					
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓓ①	读取结果数据数	对 ⓓ⓪ [2] 中指定的数据数，设置实际读取的数据数。值的单位根据数据类型指定。	0 ～ 480	系统	
	ⓓ① [1] ～ ⓓ① +□	读取数据	读取的数据	0000_H ～ $FFFF_H$	系统	
ⓓ②	通过处理结束置为 ON 的位软元件（但是异常结束时ⓓ② [1] 也置为 ON。）					位
	软元件	项目	内容以及设置数据	设置范围	设置侧	
	ⓓ② [0]	结束信号	表示处理结束。 ON：结束 OFF：未结束	－	系统	
	ⓓ② [1]	异常结束信号	表示是正常结束还是异常结束。 ON：异常结束 OFF：正常结束	－		

注意事项

(1) QCPU(Q 模式) 的情况下，ⓢ0 (驱动器指定) 中只能指定 ATA 卡的驱动器 (2)。
但是安装了 Flash 卡的情况下，不能通过 SP_FWRITE 指令进行写入。
不能设置为 SRAM 卡、标准 RAM、标准 ROM 的驱动器。
LCPU 的情况下，ⓢ0 (驱动器指定) 中只能设置 SD 存储卡的驱动器 (2)。

(2) CSV 设置时被写入的数据为 10 进制数的值。
例 字符 “A”(41H) → 65 被写入。
可用范围 : -32768 ～ 32767

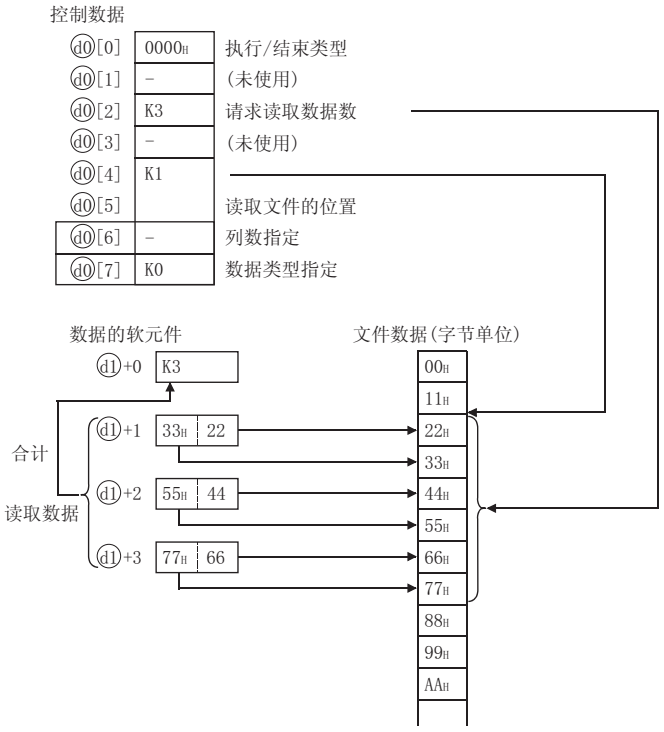
(3) 二进制读取时，字指定中文件位置的设置范围为 00000000H ～ 7FFFFFFFH。

功 能

(1) 从指定文件中读取数据。
根据控制数据的执行 / 结束类型，指定将文件的内容是以二进制数据读取，还是将文件的内容转换为 CSV 格式后进行读取。(QCPU(Q 模式) 的情况下读取的对象只能为 ATA 卡，LCPU 的情况下读取的对象只能为 SD 存储卡。)

(2) 对于处理结束ⓓ2 位软元件，在检测到本指令的处理结束的 END 执行时将自动地置为 ON，通过下一个扫描的 END 置为 OFF。
因此作为本指令的执行结束标志使用。
本指令异常结束时，异常结束ⓓ2 [1] 的软元件与处理结束ⓓ2 [0] 的软元件以相同的时机 ON/OFF，因此将ⓓ1 [1] 作为本指令的异常结束标志使用。
此外，指令执行过程中 SM721 将被置为 ON。
SM721 处于 ON 的状态下，不能执行本指令。(执行的情况下将成为无处理。)
再者，在执行指令时检测出出错 (SM721 为 ON 之前) 的情况下，处理结束 [0]、异常结束ⓓ1 [0] 以及 SM721 不变为 ON。

(3) 数据的请求读取数据数ⓓ0 [2]、文件位置ⓓ0 [4]，ⓓ0 [5] 以及读取数据软元件容量的处理单位是以字单位进行指定。
二进制读取的情况下，各个指定情况下的数据读取方法如下所示。



(4) 二进制读取时

(a) 省略了对象文件的扩展名的情况下，扩展名将变为 “.BIN”。

(b) 指定了不存在的文件的情况下，将变为出错状态。

(c) 指定了大于已有文件容量的文件位置时的情况如下所示。

- 对于序列号的前 5 位数为 “01111”以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 通用型 QCPU，将变为读取 0 点并正常结束。

(5) CSV 格式转换读取时

(a) 将 CSV 格式文件的要素 (EXECL 的单元格) 按行方向的顺序进行读取，将数值字符串转换为二进制值后存储到软元件中。

(b) 在省略了扩展名的情况下，扩展名将变为 “.CSV”。

(c) 指定了不存在的文件的情况下将变为出错状态。

(d) 从文件的起始开始请求读取数据数 ⑩ [2] 中指定的要素将被读取。
在对指定数据数的数据进行读取之前，到达了文件的最终数据时的情况如下所示。

- 对于序列号的前 5 位数为 “01111”以前的高性能型 QCPU，将变为出错状态。
- 对于序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 通用型 QCPU，对可读取的数据进行读取。

(e) 指定列数为 0 的情况下，将在忽略 CSV 格式文件的行分割的状况下进行读取。

例 CSV 格式转换读取时指定列数为 0 的情况下

通过EXCEL创建的数据

	A	B	C
1	Main/sub item		Measured value
2	Length	1	3
3	Temperature	-21	
4			

以CSV格式保存的数据

大/小项目	,	,	测定值	CR	LF
长度	,	1	, 3	CR	LF
温度	,	-21	,	CR	LF

读取到软元件中的数据

```
SP_FREAD(TRUE,2,Var_D20,Var_D10,Var_D99,Var_M0);
                                  → 被读取的数据
                        → 控制数据
              → 文件名
```

控制数据

Var_D10	0100H	执行/结束类型
	-	未使用
	K9	读取数据数请求
	-	未使用
	K0	
	K0	
	K0	列数指定
	K0	数据类型指定
Var_D20	4241H	文件名
	4443H	"ABCDE"
	0045H	

被读取的数据

存储读取的数据

	Var_D99	K9	读取结果数据数
读取数据	大/小项目	K0	由于“大/小项目” 是数字以外的数据，因此存储转换数据(0)。
	,与,之间的数据	K0	由于“ ”是数值以外的数据，因此存储转换数据(0)。
	测定值	K0	由于“测定值”是数值以外的数据，因此存储转换数据(0)。
	长度	K0	由于“长度”是数值以外的数据，因此存储转换数据(0)。
	1	K1	由于1是数值，因此被转换为二进制值。
	3	K3	由于3是数值，因此被转换为二进制值。
	温度	K0	由于“温度”是数值以外的数据，因此存储转换数据(0)。
	K-21	K-21	由于-21是数值，因此被转换为二进制值。
	,与CR之间的数据	K0	由于“ ”是数值以外的数据，因此存储转换数据(0)。

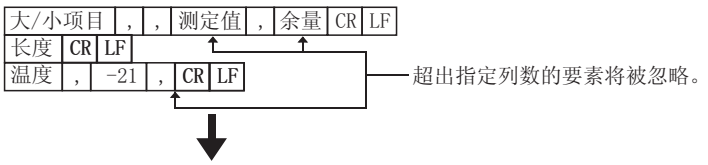
即使在各行的列数不同的情况下，也将在忽略行的状况下进行读取。

要 点

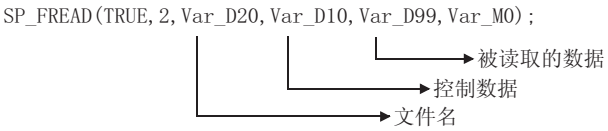
在 EXCEL 中不会创建出这样的文件。在用户对 CSV 文件进行了修正的情况下会引起此现象。

例 读取时各行的列数不相同的情况下

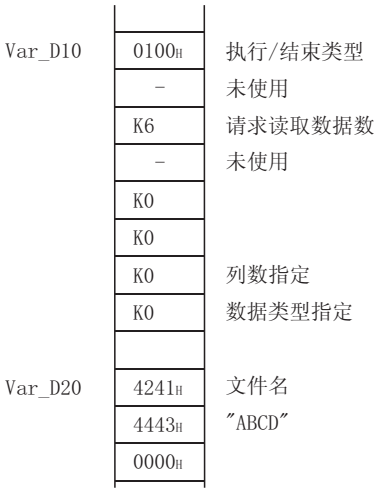
以CSV格式保存的数据



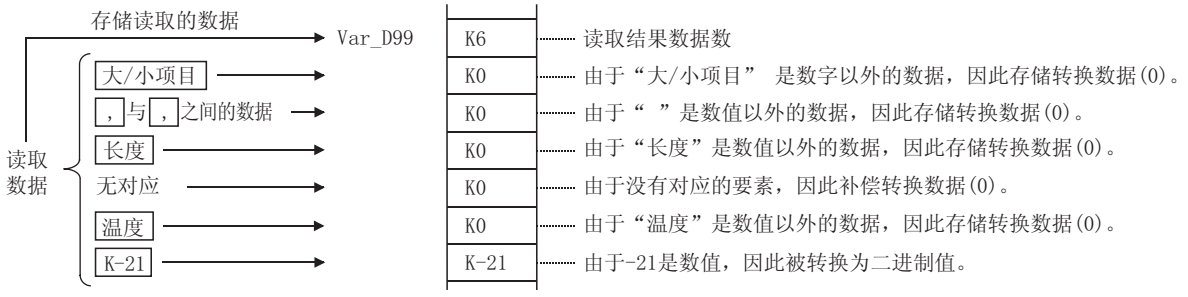
被读取到软元件中的数据



控制数据



被读取的数据



(f) 在 CSV 格式转换读取中，指定列数为 0 以外的情况下，将被作为指定列数的表格的 CSV 格式文件进行读取。超出指定列数的要素将被忽略。

例 CSV 格式转换读取时指定列数为 0 以外的情况下

通过EXCEL创建的数据

	A	B	C
1	Main/sub item		Measured value
2	Length	1	3
3	Temperature	-21	

以CSV格式保存的数据

大/小项目	,	,	测定值	CR	LF	
长度	,	1	,	3	CR	LF
温度	,	-21	,		CR	LF

超出指定列数的要素被忽略。

被读取到软元件中的数据

```
SP_FREAD(TRUE,2,Var_D20,Var_D10,Var_D99,Var_M0);
                                  被读取的数据
                          控制数据
                  文件名
```

控制数据

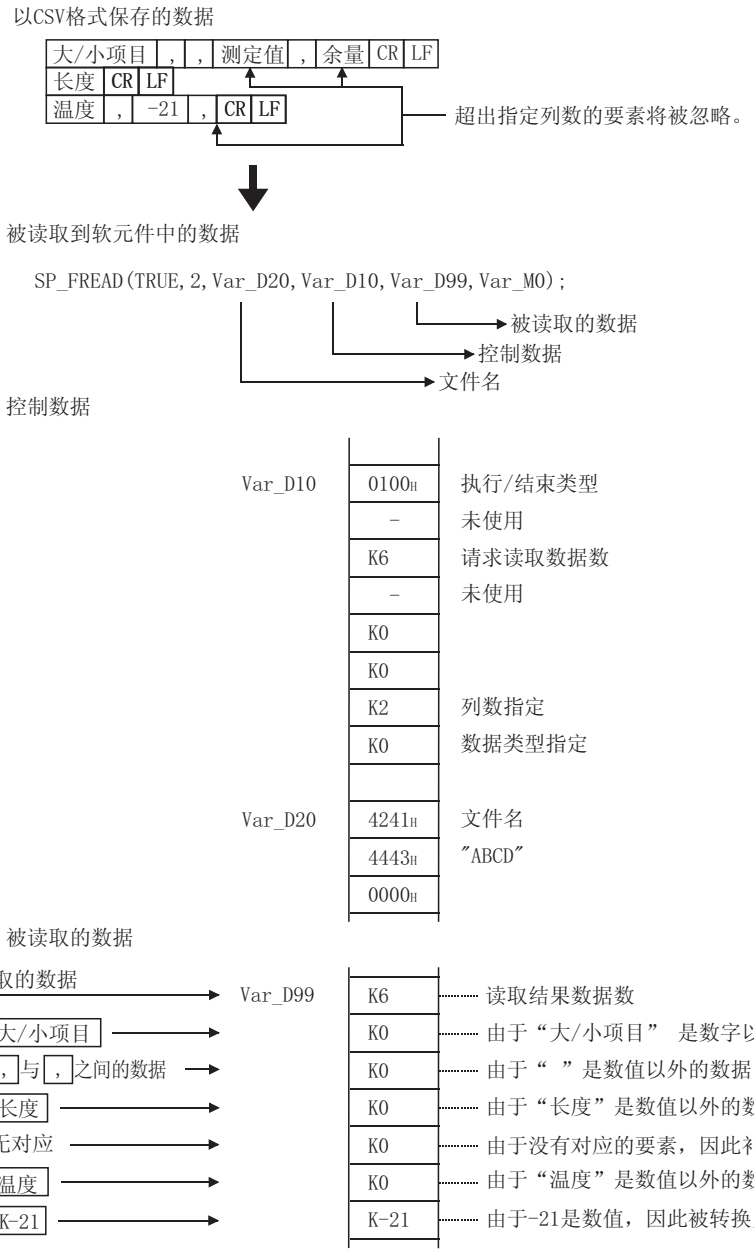
Var_D10	0100_H	执行/结束类型
	-	未使用
	K6	请求读取数据数
	-	未使用
	K0	
	K0	
	K2	列数指定
	K0	数据类型指定
Var_D20	4241_H	文件名
	4443_H	"ABCD"
	0000_H	

被读取的数据

存储读取的数据	Var_D99	K6	读取结果数据数
读取数据 大/小项目		K0	由于“大/小项目” 是数字以外的数据，因此存储转换数据(0)。
,与,之间的数据		K0	由于“ ”是数值以外的数据，因此存储转换数据(0)。
长度		K0	由于“长度”是数值以外的数据，因此存储转换数据(0)。
1		K1	由于1是数值，因此被转换为二进制值。
温度		K0	由于“温度”是数值以外的数据，因此存储转换数据(0)。
K-21		K-21	由于-21是数值，因此被转换为二进制值。

即使在各行的列数不相同的情况下，超出指定列数的要素将被忽视，指定列数不足的列将被补偿0。

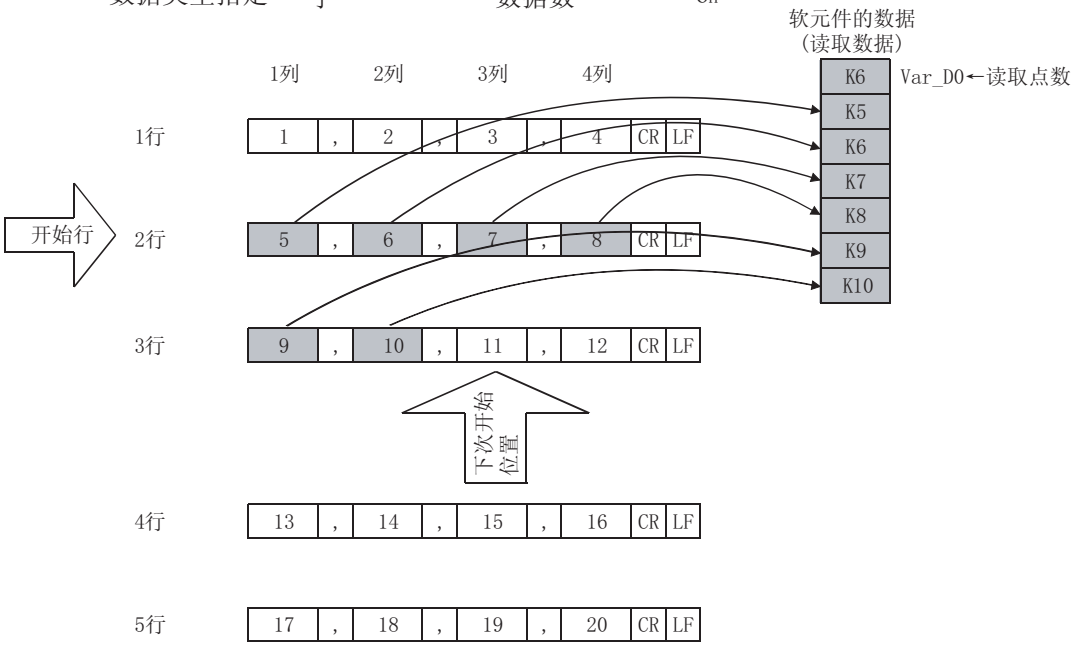
例 读取时各行的列数不相同的情况下



(g) 在序列号的前 5 位数为 “01112”以后的高性能型 QCPU/ 通用型 QCPU 中，可以分为多次进行读取。

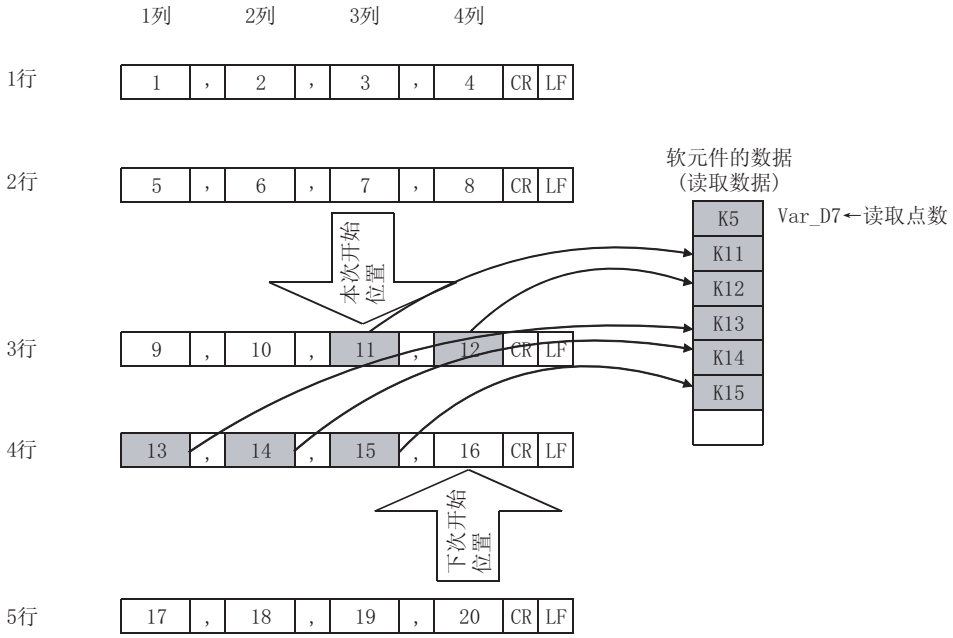
[指定希望开始读取的行。]

执行类型 = CSV格式　　开始行数 = 2H
列指定 = 4H　　读取起始软元件 = Var_D0
数据类型指定 = 字　　数据数 = 6H



[通过上次继续模式，从上次读取位置继续进行读取。]

执行类型 = CSV格式　　开始行数 = FFFFFFFFH(继续模式)
列指定 = 4H　　读取起始软元件 = Var_D7
数据类型指定 = 字　　数据数 = 5H



- 通过继续模式进行读取的情况下，如果将 “执行类型”、“列指定”、“数据类型指定”设置得与上次的不相同，将无法正常地进行至上次的添加。
- 在通过继续模式继续读取数据的过程中，如果执行其它设置的 SP_FREAD 指令及 SP_FWRITE 指令，将无法正常地进行至上次的添加。

(h) 在 CSV 格式转换读取中，读取的 CSV 格式文件中存在有超出范围的数值或数值以外的要素的情况下，将被转换为 0H。

(i) 在 CSV 格式转换读取中，读取时的数值转换如下所示。

CSV 内数值		-32768 ～ -1	0 ～ 32767	32768 ～ 65535
字软元件	无符号	32768 ～ 65535	0 ～ 32767	32768 ～ 65535
	有符号	-32768 ～ -1	0 ～ 32767	-32768 ～ -1

(j) 在中断程序中不要执行本指令。
（在中断程序执行时将导致误动作。）

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- 驱动器指定ⓢ⓪中指定的驱动器为 ATA 卡以外时。
  (QCPU(Q 模式）时）（出错代码： 4100)
- 驱动器指定ⓢ⓪中指定的驱动器为 SD 存储卡以外时。(LCPU 时）（出错代码： 4100)
- 控制数据ⓓ⓪以后中设置的值超出了设置范围时。（ⓓ⓪ [2] 除外）（出错代码： 4100)
- 读取数据数ⓓ⓪ [2] 中指定的值超出了设置范围时。（出错代码： 4101)
- 指定了不能指定的软元件时。（出错代码： 4004)
- 文件名字符串ⓓ①以后中指定的文件名不存在于指定的驱动器中时。
  （出错代码： 2410)
- 读取的数据的容量超出了读取软元件容量时。（出错代码： 4101)
- 二进制读取时与请求读取数据数ⓓ⓪ [2] 中指定的容量相比，文件的数据数不足时。
  （序列号的前 5 位数为 “01111” 以前的高性能型 QCPU）（出错代码： 4100)
- ATA 卡内发生了访问异常时。(QCPU(Q 模式）时）（出错代码： 4100)
- SD 存储卡内发生了访问异常时。(LCPU 时）（出错代码： 4100)
- ⓓ⓪或ⓓ①中指定的软元件超出了相应软元件的范围时。
  （通用型 QCPU、LCPU 时）（出错代码： 4101)

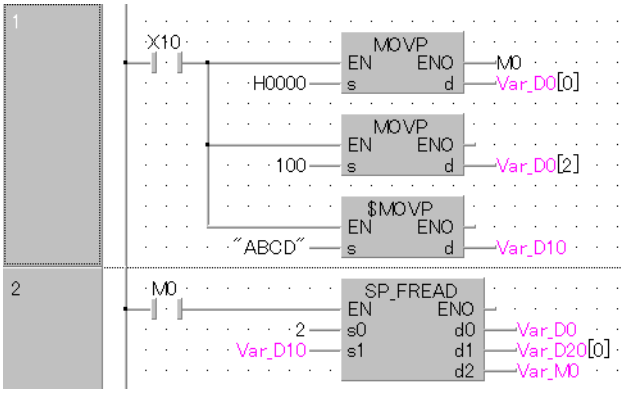
程序示例

(1) 以下为将 X10 置为 ON 时，从驱动器 2 中安装的存储卡的文件 “ABCD.BIN”的起始开始，以二进制读取 4 字节的程序。

• 对于控制数据用软元件，假设已预留了 Var_D0 算起的 8 点。

• 对于读取用软元件，假设已预留了 Var_D20 算起的 100 字节。

[结构体梯形图]




```
[ST]
IF X10 THEN
    MOVP(TRUE,H0000,Var_D0[0]);
    MOVP(TRUE,100,Var_D0[2]);
    Var_D10:="ABCD";
    SP_FREAD(TRUE,2,Var_D10,Var_D0,Var_D20[0],Var_M0);
END_IF;
```

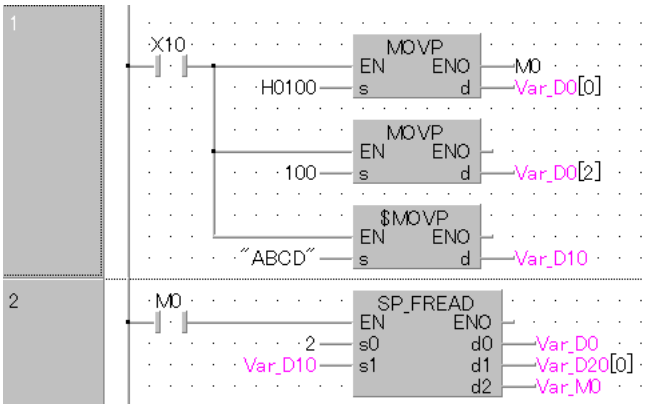
(2) 以下为将 X10 置为 ON 时，将文件名 “ABCD.CSV”以 2 列的 CSV 格式读取到插槽 0 中安装的可编程控制器卡中的程序。

• 对于控制数据用软元件，假设已预留了 Var_D0 算起的 8 点。

• 对于读取用软元件，假设已预留了 Var_D20 算起的 100 字节。

• 假设读取的 CSV 格式文件中，不存在数值以外的要素。

[结构体梯形图]



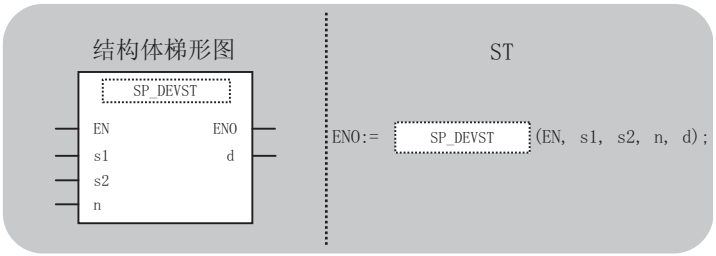

```
[ST]
IF X10 THEN
    MOVP(TRUE,H0100,Var_D0[0]);
    MOVP(TRUE,100,Var_D0[2]);
    Var_D10:="ABCD";
    SP_FREAD(TRUE,2,Var_D10,Var_D0,Var_D20[0],Var_M0);
END_IF;
```

7.18.14 至标准 ROM 的数据写入

SP_DEVST

Basic | High performance | Universal | L CPU

SP_DEVST



中放入下述指令。

SP_DEVST

输入自变量，EN: 执行条件 : 位

s1: 软元件存储用文件的写入偏置（以 1 点 16 位的单位指定） : ANY32

s2: 写入到标准 ROM 中的软元件的起始编号 : ANY16

n: 写入软元件点数 : ANY16

输出自变量，ENO: 执行结果 : 位

d: 通过执行结束置为 ON 的数组 : 位的数组 (0..1)

- d[0]: 结束
- d[1]: 异常结束

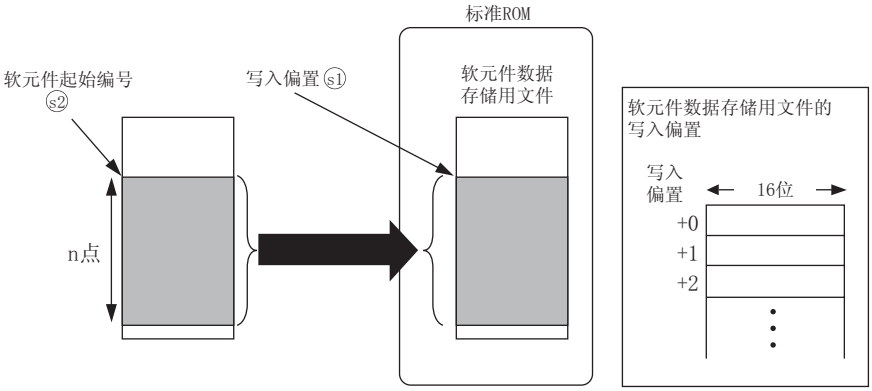
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
⑤1	-	○	○	-				○	-
⑤2	-	○	○	-				-	-
n	-	○	○	-				○	-
ⓓ	△ *1	-	△ *1	-				-	-

*1 ：不能使用局部软元件以及各程序中设置的文件寄存器。

功 能

(1) 将⑤2 中指定的软元件的 n 中指定的点数的软元件数据，写入到标准 ROM 上的软元件数据存储用文件的⑤1 中指定的写入偏置中。

⑤1 是从软元件数据存储用文件起始处开始的偏置，通过字偏置（每 16 位进行 +1 的单位）进行指定。



(2) 对于标准 ROM 的软元件数据写入位置结束 ⓓ [0]，在检测到本指令的处理结束的 END 指令执行时将自动地置为 ON，通过下一个扫描的 END 指令置为 OFF，因此作为本指令的执行结束标志使用。

(3) 本指令异常结束时，异常结束 ⓓ [1] 与结束 ⓓ [0] 以相同的时机置为 ON/OFF，因此将 ⓓ [1] 作为本指令的异常结束标志使用。

(4) 本指令的执行过程中 SM721 将被置为 ON。
SM721 已处于 ON 状态的情况下，不能执行本指令。（执行的情况下将成为无处理。）

(5) 在执行指令时检测出出错的情况下，结束 ⓓ [0]、异常结束 ⓓ [1]、SM721 不变为 ON。

出 错

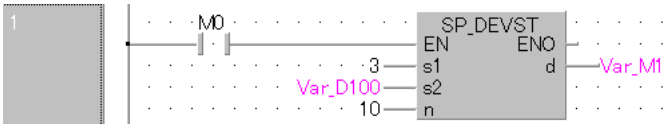
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ1 中指定的写入偏置超出了软元件数据存储用文件的范围时。
（出错代码： 4100)
- ⓢ1 中指定的写入偏置算起的 n 点超出了软元件数据存储用文件的范围时。
（出错代码： 4100)
- ⓢ2 中指定的软元件算起的 n 点的范围超出了相应软元件时。 （出错代码： 4101)
- 可编程控制器参数的可编程控制器文件设置中，未进行软元件存储用文件的设置时。
（出错代码： 2410)
- ⓓ 中指定的软元件超出了相应软元件的范围时。 （出错代码： 4101)

程序示例

以下为 M0 变为 ON 时，将 Var_D100 算起的 10 点写入到标准 ROM 的软元件数据存储用文件中的程序。

[结构体梯形图]



[ST]

```
SP_DEVST(M0,3,Var_D100,10,Var_M1);
```

注意事项

(1) 写入到标准 ROM 的值将成为执行本指令时的值。

(2) 随着 SP_DEVST 指令的执行，标准 ROM 写入次数指标 (SD687、SD688) 将增加。
标准 ROM 写入次数指标超过了 10 万次时，将变为 FLASH ROM ERROR(出错代码 : 1610) 状态。

(3) 为了防止由于意外的指令执行导致的 ROM 写入次数的增加，对至标准 ROM 的写入指令执行次数进行指定 (SD695)，对 1 日中的写入次数进行限制。
超过了设置的写入次数 (默认值 : 36 次) 时，将变为 OPERATION ERROR(出错代码 : 4113) 状态。

7.18.15 从标准 ROM 的数据读取

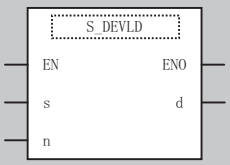
S_DEVLD

Basic | High performance | Universal | L CPU

S(P)_DEVLD

P：执行条件　　：

结构体梯形图



ST

```
ENO:= S_DEVLD (EN, s, n, d);
```

[ ] 中放入下述指令。

S_DEVLD　　SP_DEVLD

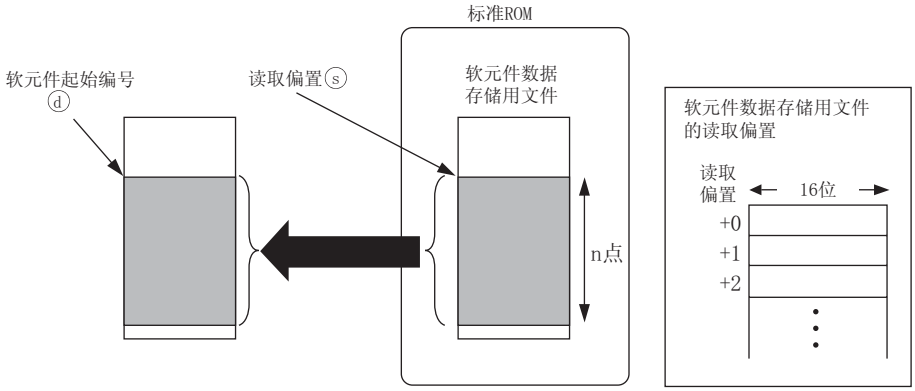
输入自变量，EN:　执行条件　：位
　　s:　软元件存储用文件的读取偏置（以 1 点 16 位单位进行指定）　：ANY32
　　n:　读取软元件点数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　标准 ROM 读取软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
ⓢ	-	○		_				○	-
n	-	○		_				○	-
ⓓ	-	○		_				-	-

功 能

从标准 ROM 上的软元件数据存储用文件的ⓢ中指定的读取偏置开始，对 n 中指定点数的软元件数据进行读取后，存储到ⓓ中指定的软元件中。

ⓢ是从软元件数据存储用文件起始处开始的偏置，通过字偏置（每 16 位进行 +1 的单位）进行指定。



出 错

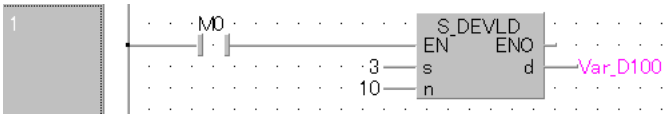
在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的地址超出了标准 ROM 的范围时。（出错代码： 4100）
- ⓢ中指定的地址算起的 n 点超出了标准 ROM 的范围时。（出错代码： 4100）
- ⓓ中指定的软元件算起 n 点的范围超出了相应软元件时。（出错代码： 4101）
- 可编程控制器参数的可编程控制器文件设置中，未进行软元件存储用文件的设置时。（出错代码： 2410）

程序示例

以下为 M0 变为 ON 时，从标准 ROM 的软元件数据存储用文件中将 10 点的数据读取到 Var_D100 中的程序。

[结构体梯形图]



[ST]

```
S_DEVLD(M0,3,10,Var_D100);
```

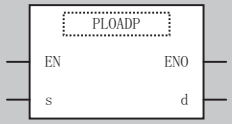
7.18.16 从存储卡的程序装载

PLOADP

Basic | High performance | Universal | L CPU

PLOADP

结构体梯形图



ST

```
ENO:= PLOADP (EN, s, d);
```

中放入下述指令。

PLOADP

输入自变量， EN: 执行条件 : 位
s: 存储装载程序的驱动器 No. 及文件名*1 : 字符串
输出自变量， ENO: 执行结果 : 位
d: 指令结束时置为 1 个扫描 ON 的软元件 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
ⓢ	-	○		-				○	-
ⓓ	△*2	-		-				-	-

*1 ：以 “〈驱动器编号〉： 〈文件名〉”进行指定。例 1: MAIN
*2 ：不能使用局部软元件。

功 能

(1) 将存储卡、标准 ROM 中存储的程序，传送至程序存储器（驱动器 0）中。
即使传送的程序在可编程控制器参数的程序设置中未被登录，也将通过 CPU 模块内部的程序设置将其置为待机类型。
此时可编程控制器参数的程序设置不变化。
（通过 PLOADP 指令传送程序的情况下，程序存储器中需要有连续的空余区域。）

(2) 对于通过 PLOADP 指令添加的程序，将被分配未使用程序 No. 中最小的编号。
（由用户指定的情况下，应将程序 No. 存储到 SD720 中。）
例如，通过 PLOADP 指令添加了 “MAIN6”的情况如下所示。

(a) 在所设置的程序 No. 已被占满的情况下，将被添加到所设置的程序 No. 的最后处。
程序 No.1 ～ 5 中已被设置了程序的情况下，将增添新的程序 No.。

程序No.	程序名
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

通过PLOADP指令添加“MAIN6”。

程序No.	程序名
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5
6	MAIN6

←添加在最后处。

(b) 在存在有多个未放入程序的空余程序 No. 的情况下，PLOADP 指令中指定的程序将被添加到最小号的空余程序 No. 中。
（通过 PUNLOADP 指令删除了程序时将会产生空余的程序 No.。）
存在有空余的程序 No.2 及 4 的情况下，将被添加到程序 No.2 中。

程序No.	程序名
1	MAIN1
2	空余
3	MAIN3
4	空余
5	MAIN5

通过PLOADP指令添加“MAIN6”。

程序No.	程序名
1	MAIN1
2	MAIN6
3	MAIN3
4	空余
5	MAIN5

←添加到空余的最小号中。

(3) 可以指定驱动器 No.1、2、4。（不能指定驱动器 3）
- 驱动器 1：存储卡 (RAM)
- 驱动器 2：存储卡 (ROM)
- 驱动器 4：标准 ROM

(4) 文件名中不需要指定扩展名 (.QPG)。

(5) 在本指令结束的扫描的 END 处理中，将ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(6) 不能同时执行 PLOADP/PUNLOADP/PSWAPP 指令。
同时执行了多个上述指令的情况下，后执行的指令将变为非执行。
使用上述指令时，用户应采取互锁以防止同时执行。

(7) 在中断程序中不要执行本指令。
（ 在中断程序执行时将导致误动作。）

(8) 执行通过本指令传送至程序存储器中的程序的情况下，应通过 PSCAN 指令置为 “扫描执行类型”。（参阅 7.17.3 项）

(9) 通过本指令传送的程序的可编程控制器文件设置如下所示。

(a) 各程序文件的使用方法
通过本指令传送的程序的文件寄存器、软元件初始值、注释、局部软元件的使用方法全部为“按照可编程控制器文件设置”。
但是，通过本指令传送程序时，如果满足下述两个条件将变为出错状态。
• 在可编程控制器文件设置中设置为“使用局部软元件”。
• 程序存储器的程序个数超过了参数中设置的程序个数时。

在通过本指令传送的程序中使用局部软元件的情况下，应在参数中登录虚拟的程序文件。然后，应将虚拟的程序文件通过 PUNLOADP 指令删除后，通过 PLOADP 指令进行装载。

(b) I/O 刷新设置
对于通过本指令传送的程序的 I/O 刷新设置，输入、输出均为“无设置”。

(10)“PLOADP 指令”与“运行中写入”的处理不能同时执行。

(a) 在 PLOADP 指令的处理过程中，存在有运行中写入请求时，运行中写入将等待。
PLOADP 指令的处理结束后，执行运行中写入。

(b) 在运行中写入时如果执行 PLOADP 指令，PLOADP 指令的处理将等待。
运行中写入结束后，进行 PLOADP 指令的处理。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

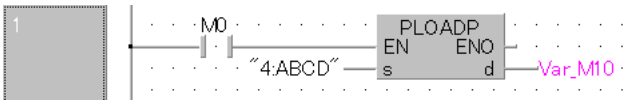
- ⓢ 中指定的驱动器 No. 或文件名不存在时。 (出错代码： 2410)
- ⓢ 中指定的驱动器 No. 为不能指定的驱动器时。 (出错代码： 4100)
- 驱动器 0 中未能预留用于装载相应程序的容量时。 (出错代码： 2413)
- 程序存储器中已登录了下述表格中的文件格数时。 (出错代码： 4101)
- SD720 中存储的程序 No. 已被使用，或超出了下表中的程序 No. 的范围时。 (出错代码： 4101)
- 存在有与要装载的程序文件相同名称的程序文件时。 (出错代码： 2410)
- 未能预留出局部软元件的文件容量时。 (出错代码： 2401)

CPU 型号	程序存储器 (文件个数)	最大程序 No.
Q02(H)CPU	28 个	28
Q06HCPU	60 个	60
Q12HCPU	124 个	124
Q25HCPU	124 个	124

程序示例

以下为 M0 变为 ON 时，将驱动器 4 中存储的 “ABCD.QPG”传送至驱动器 0 中后，置为待机状态的程序。

[结构体梯形图]



[ST]

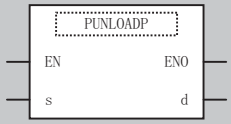
```
PLOADP(M0,"4:ABCD",Var_M10);
```

7.18.17 从程序存储器中的程序卸载

PUNLOADP

PUNLOADP

结构体梯形图



ST

```
ENO:= PUNLOADP (EN, s, d);
```

[ ] 中放入下述指令。

PUNLOADP

输入自变量， EN: 执行条件 : 位
s: 卸载的程序文件名 : 字符串
输出自变量， ENO: 执行结果 : 位
d: 指令结束时置为 1 个扫描 ON 的软元件 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
ⓢ	-	○		-				○	-
ⓓ	△*1	-		-				-	-

*1 ：不能使用局部软元件。

功 能

(1) 将程序存储器（驱动器 0）中存储的待机程序从程序存储器中删除。
对于通过 PSCAN 指令置为 “扫描执行类型”的程序或通过 PLOW 指令置为 “低速执行类型”的程序不能删除。

(2) 通过 PUNLOADP 指令删除的程序 No. 将变为 “空余”。
在可编程控制器参数的程序设置中设置了程序 No.1 ～ 5 时，如果通过本指令将程序 No.2 的程序删除，程序 No.2 将变为空余。

程序No.	程序名
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

通过PUNLOADP指令删除“MAIN2”。

程序No.	程序名
1	MAIN1
2	空余
3	MAIN3
4	MAIN4
5	MAIN5

←程序No.2被删除。

(3) 文件名中不需要指定扩展名 (.QPG)。

(4) 在本指令结束的扫描的 END 处理中，将ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(5) 不能同时执行 PLOADP/PUNLOADP/PSWAPP 指令。
同时执行了多个上述指令的情况下，后执行的指令将变为非执行。
使用上述指令时，用户应采取互锁以防止同时执行。

(6) 执行 PUNLOADP 指令后，将可编程控制器 CPU 的电源进行了 OFF → ON 或对 CPU 模块进行了复位时的情况如下所示。

(a) 在可编程控制器参数中进行了引导设置的情况下，进行了引导设置的程序将被传送到程序存储器中。
不执行通过 PUNLOADP 指令删除的程序的情况下，应从可编程控制器参数的引导设置及程序设置中将相应的程序名删除。

(b) 在可编程控制器参数中未进行引导设置的情况下，将变为 “FILE SET ERROR(出错代码 2400)”状态。

1) 不执行通过 PUNLOADP 指令删除的程序的情况下，应从可编程控制器参数的程序设置中将相应的程序名删除。

2) 若要再次执行通过 PUNLOADP 指令删除的程序，应将相应的程序写入到 CPU 模块中。

(7) 在中断程序中不要执行本指令。
(在中断程序执行时将导致误动作。)

(8) 对于通过本指令从程序存储器中删除的程序，应事先通过 PSTOP 指令将其置为 “待机执行类型”。(参阅 7.17.1 项)

(9) “PUNLOADP 指令”与 “运行中写入”的处理不能同时执行。

(a) 在 PUNLOADP 指令的处理过程中，存在有运行中写入请求时，运行中写入将等待。
PUNLOADP 指令的处理结束后，执行运行中写入。

(b) 在运行中写入时如果执行 PUNLOADP 指令，PUNLOADP 指令的处理将等待。
运行中写入结束后，进行 PUNLOADP 指令的处理。

出 错

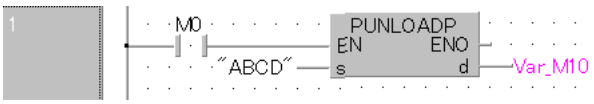
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ中指定的驱动器 No. 的文件名不存在时。 (出错代码 : 2410)
- ⓢ中指定的程序不是待机程序，或是当前正在执行中的程序时。 (出错代码 : 4101)

程序示例

以下为 M0 由 OFF → ON 时，将驱动器 0 中存储的 “ABCD.QPG”从存储器中删除的程序。

[结构体梯形图]



[ST]

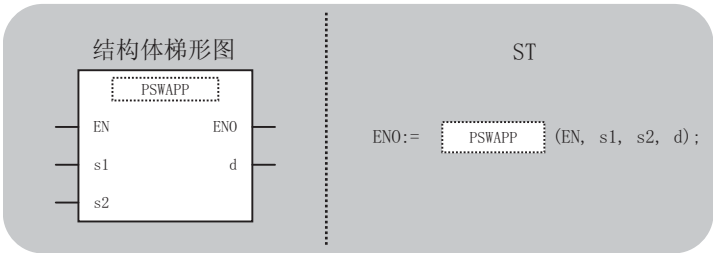
```
PUNLOADP(M0,"ABCD",Var_M10);
```

7.18.18 装载 + 卸载

PSWAPP

Basic | High performance | Universal | L CPU

PSWAPP



[ ] 中放入下述指令。

PSWAPP

输入自变量， EN: 执行条件 : 位
s1: 卸载的程序的文件名 : 字符串
s2: 存储装载的程序的驱动器 No. 文件名 *1 : 字符串
输出自变量， ENO: 执行结果 : 位
d: 指令结束时置为 1 个扫描 ON 的软元件 : 位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 $	其它数据
	位	字		位	字				
ⓢ1	–	○		–				○	–
ⓢ2	–	○		–				○	–
ⓓ	△ *2	–		–				–	–

*1 ：以 “〈驱动器编号〉： 〈文件名〉” 进行指定。例 1: MAIN
*2 ：不能使用局部软元件。

功 能

(1) 将ⓢ1 中指定的程序存储器（驱动器 0）中存储的待机类型程序从程序存储器中删除。然后，将ⓢ2 中指定的存储卡、标准 ROM 中存储的程序传送至程序存储器中，置为待机状态。
（将程序传送至程序存储器中的情况下，程序存储器需要有连续的空余区域。）
此外，通过 PSCAN(P) 指令置为 “扫描执行类型” 的程序或通过 PLOW(P) 指令置为 “低速执行类型” 的程序不能删除。

(2) 对于通过 PSWAPP 指令传送至程序存储器中的程序，将变为从程序存储器中删除的程序的程序 No.。

(即使从程序存储器中删除的程序的程序 No. 的前面存在有空余的号，也不变为空余的程序 No.。)

程序 No.2 为 “空余”时，如果通过本指令对程序 No.3 的程序进行替换，传送至程序存储器中的程序将被登录为程序 No.3。

程序No.	程序名
1	MAIN1
2	空余
3	MAIN3
4	MAIN4
5	MAIN5

通过PSWAPP指令对“MAIN3”与“MAIN6”进行替换。

程序No.	程序名	
1	MAIN1	
2	空余	
3	MAIN6	←变为MAIN6。
4	MAIN4	
5	MAIN5	

(3) 驱动器 No. 可指定为 1、2、4。(不能指定为驱动器 3。)

- 驱动器 1: 存储卡 (RAM)
- 驱动器 2: 存储卡 (ROM)
- 驱动器 4: 标准 ROM

(4) 文件名中不需要指定扩展名 (.QPG)。

(5) 在本指令的执行结束的扫描的 END 处理中，将ⓓ中指定的位软元件置为 ON，在下一个 END 处理中置为 OFF。

(6) 不能同时执行 PLOADP/PUNLOADP/PSWAPP 指令。

同时执行了多个上述指令的情况下，后执行的指令将变为非执行。
使用上述指令时，用户应采取互锁以防止同时执行。

(7) 执行 PSWAPP 指令后，将可编程控制器 CPU 的电源进行了 OFF → ON 或对 CPU 模块进行了复位时的情况如下所示。

(a) 在可编程控制器参数中进行了引导设置的情况下，进行了引导设置的程序将被传送到程序存储器中。

执行通过 PSWAPP 指令替换的程序时，应将可编程控制器参数的引导设置及程序设置变更为相应的程序名。

(b) 在可编程控制器参数中未进行引导设置的情况下，将变为 “FILE SET ERROR(出错代码 2400)”状态。

1) 执行通过 PSWAPP 指令替换的程序的情况下，应将可编程控制器参数的程序设置变更为相应的程序名。
2) 执行在可编程控制器参数的程序设置中设置的程序的情况下，应将相应程序再次写入到 CPU 模块中。

(8) 在中断程序中不要执行本指令。

(在中断程序执行时将导致误动作。)

(9) 执行了 PSWAPP 指令的程序的可编程控制器文件设置如下所示。

(a) 各程序文件的使用方法

执行了 PSWAPP 指令后的程序的文件寄存器、软元件初始值、注释、局部软元件的使用方法全部为 “按照可编程控制器文件设置”。

(b) I/O 刷新设置

对于执行了 PSWAPP 指令后的程序的 I/O 刷新设置，输入、输出均为 “无设置”。

(10)“PSWAPP 指令”与 “运行中写入”的处理不能同时执行。

(a) 在 PSWAPP 指令的处理过程中，存在有运行中写入请求时，运行中写入将等待。
PSWAPP 指令的执行结束后，执行运行中写入。

(b) 运行中写入中时如果执行 PSWAPP 指令，PSWAPP 指令的处理将等待。
运行中写入结束后，进行 PSWAPP 指令的处理。

出 错

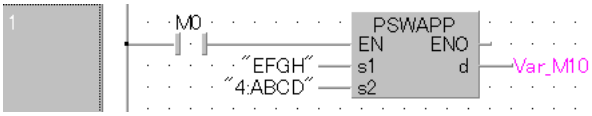
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ①, ② 中指定的驱动器 No. 或文件名不存在时。 (出错代码： 2410)
- ① 中指定的驱动器 No. 为不能指定的驱动器时。 (出错代码： 4100)
- 驱动器 0 中未能预留用于装载相应程序的容量时。 (出错代码： 2413)
- ① 中指定的程序不是待机类型程序，或者是正在执行中的程序时。(出错代码： 4101)

程序示例

以下为 M0 由 OFF → ON 时，在将驱动器 0 中存储的 “EFGH.QPG”从存储器中删除的同时，将驱动器 4 中存储的 “ABCD.QPG”传送至驱动器 0 中，置为待机状态的程序。

[结构体梯形图]



[ST]

```
PSWAPP(M0,"EFGH","4:ABCD",Var_M10);
```

7.18.19 文件寄存器高速块传送

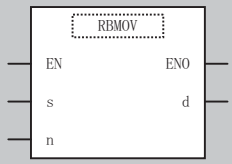
RBMOV

Basic　High performance　Universal　L CPU

RBMOV(P)

P：执行条件　　：

结构体梯形图



ST

ENO:= RBMOV (EN, s, n, d);

中放入下述指令。

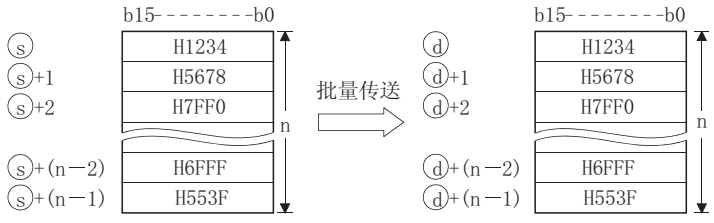
RBMOV　　RBMOVP

输入自变量，EN:　执行条件　：位
　　s:　存储传送的数据的软元件的起始编号　：ANY16
　　n:　传送数　：ANY16
输出自变量，ENO:　执行结果　：位
　　d:　传送目标的软元件的起始编号　：ANY16

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K, H	其它数据
	位	字		位	字				
ⓢ			○				-		-
n			○				○		-
ⓓ			○				-		-

功 能

(1) 将ⓢ中指定的软元件算起 n 点的 16 位数据，批量传送至ⓓ中指定的软元件算起的 n 点。



(2) 传送源与传送目标的软元件重叠的情况下也可进行传送。
向软元件编号小的方向传送的情况下从ⓢ开始传送，向软元件编号大的方向传送的情况下从ⓢ+(n-1) 开始传送。

但是，进行从 R 至 ZR 或从 ZR 至 R 的传送的情况下，应按下述方式注意避免 ZR 与 R 的各个传送范围重叠。

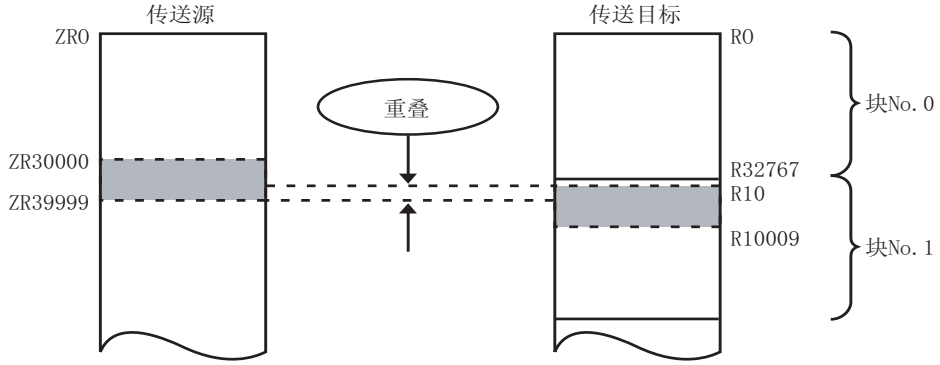
- ZR 的传送范围（(指定的 ZR 起始 No.) ～ (指定的 ZR 起始 No.+ 传送数 -1)）
- R 的传送范围（(指定的 R 起始 No.+ 文件寄存器块号×32768) ～
  (指定的 R 起始 No.+ 文件寄存器块号×32768+ 传送数 -1)）

例

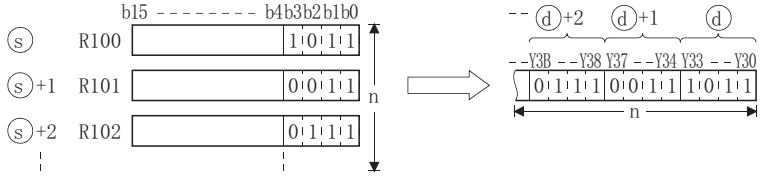
将 10000 点的数据从传送源 ZR30000 中传送至传送目标块号 1 的 R10 中的情况下，传送范围将重叠。

- ZR 的传送范围 → (30000) ～ (30000+10000-1) → (30000) ～ (39999)
- R 的传送范围 → (10+(1×32768)) ～ (10+(1×32768)+10000-1)
  → (32778) ～ (42777)

因此，从 32778 至 39999 的范围相重叠。



(3) ⓢ为字软元件而ⓓ为位软元件的情况下，对于字软元件，位软元件的位数指定中指定的位数将成为对象。
ⓓ中指定了 K1Y30 的情况下，ⓢ中指定的字软元件的低 4 位将成为对象。



出 错

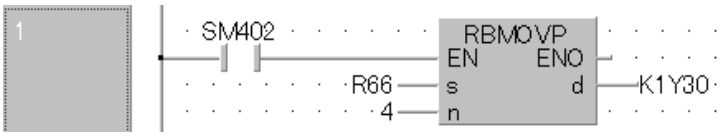
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- ⓢ，ⓓ算起的 n 点的软元件范围超出了相应软元件时。 (出错代码： 4101)
- ⓢ，ⓓ中之一未指定文件寄存器时。 (出错代码： 4101)
- n 超出了 1 ～ 10 的范围时。 (出错代码： 4100)

程序示例

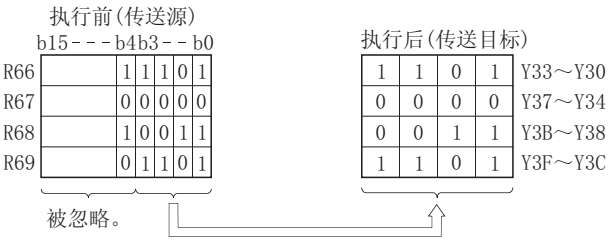
(1) 以下为将 R66 ～ R69 的低 4 位的数据以 4 点为单位从 Y30 输出到 Y3F 中的程序。

[结构体梯形图]



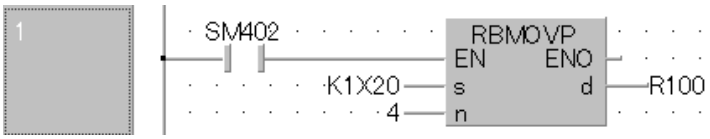
[ST]

```
RBMOVP(SM402,R66,4,K1Y30);
```



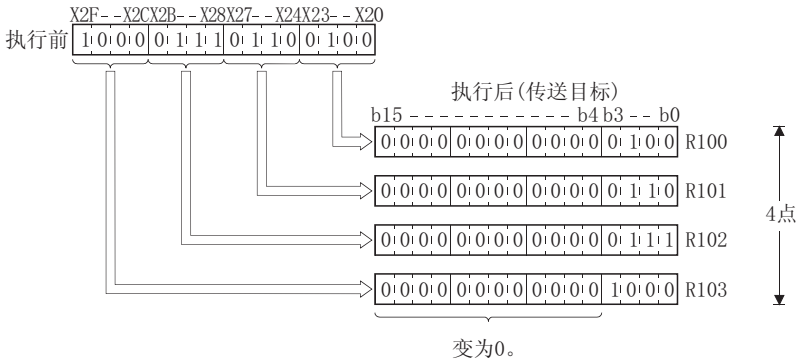
(2) 以下为将 X20 ～ X2F 的数据以 4 点为单位输出到 R100 ～ R103 中的程序。

[结构体梯形图]



[ST]

```
RBMOVP(SM402,K1X20,4,R100);
```



要 点

本指令在 QnHCPU 中对文件寄存器数据进行大批量传送的情况下有效。
在 QnUD(E)(H)CPU 的情况下，其处理速度与 BMOV 指令相同。
与 BMOV 指令的处理速度比较如下所示。
(1) 文件寄存器→内部软元件 / 内部软元件→文件寄存器的情况下。

CPU	指令	文件寄存器存储对象存储器	1 字		1000 字		10000 字	
			最小	最大	最小	最大	最小	最大
QnHCPU	RBMOV	标准 RAM	20.0 μs		91.0 μs		775.0 μs	
		SRAM 卡	22.0 μs		305.0 μs		2900.0 μs	
		Flash 卡 *1	22.5 μs		405.0 μs		3950.0 μs	
	BMOV	标准 RAM	7.5 μs		76.2 μs		720.0 μs	
		SRAM 卡	8.0 μs		384.0 μs		3900.0 μs	
		Flash 卡 *1			418.0 μs		4250.0 μs	
QnCPU	RBMOV	标准 RAM	45.5 μs		215.0 μs		1850.0 μs	
		SRAM 卡	49.5 μs		540.0 μs		5150.0 μs	
		Flash 卡 *1						
	BMOV	标准 RAM	17.5 μs		177.0 μs		1700.0 μs	
		SRAM 卡	18.0 μs		500.0 μs		5050.0 μs	
		Flash 卡 *1			572.0 μs		5800.0 μs	
Q00UCPU Q01UCPU	RBMOV	标准 RAM	12.2 μs	34.9 μs	121.5 μs	145.1 μs	1111.5 μs	1135.1 μs
		SRAM 卡 *2	–	–	–	–	–	–
		Flash 卡 *2	–	–	–	–	–	–
	BMOV	标准 RAM	7.3 μs	13.8 μs	116.5 μs	124.2 μs	1106.5 μs	1114.2 μs
		SRAM 卡 *2	–	–	–	–	–	–
		Flash 卡 *2	–	–	–	–	–	–
Q02UCPU	RBMOV	标准 RAM	9.4 μs	31.3 μs	118.5 μs	141.3 μs	1108.5 μs	1131.3 μs
		SRAM 卡	9.4 μs	31.4 μs	178.5 μs	201.3 μs	1708.5 μs	1731.3 μs
		Flash 卡 *1	9.4 μs	32.1 μs	278.5 μs	301.3 μs	2708.5 μs	2731.3 μs
	BMOV	标准 RAM	5 μs	11.6 μs	114.5 μs	122.3 μs	1104.5 μs	1112.3 μs
		SRAM 卡	5.1 μs	11.7 μs	174.5 μs	182.3 μs	1704.5 μs	1712.3 μs
		Flash 卡 *1	5 μs	11.6 μs	274.5 μs	282.3 μs	2704.5 μs	2712.3 μs
Q03UD(E)CPU	RBMOV	标准 RAM	11.3 μs	16.8 μs	120.7 μs	127.1 μs	1110.7 μs	1117.1 μs
		SRAM 卡	11.2 μs	16.7 μs	180.7 μs	187.1 μs	1710.7 μs	1717.1 μs
		Flash 卡 *1	11.3 μs	16.8 μs	280.7 μs	287.1 μs	2710.7 μs	2717.1 μs
	BMOV	标准 RAM	4.8 μs	6.6 μs	114.7 μs	117.1 μs	1104.7 μs	1107.1 μs
		SRAM 卡	4.8 μs	6.6 μs	174.7 μs	177.1 μs	1704.7 μs	1707.1 μs
		Flash 卡 *1	4.8 μs	6.5 μs	274.7 μs	277.1 μs	2704.7 μs	2707.1 μs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU	RBMOV	标准 RAM	9.2 μs	15.1 μs	61.0 μs	68.6 μs	531.0 μs	538.6 μs
		SRAM 卡	9.4 μs	15.6 μs	165.0 μs	172.6 μs	1576.0 μs	1583.6 μs
		Flash 卡 *1	9.4 μs	15.7 μs	260.0 μs	267.6 μs	2526.0 μs	2533.6 μs
	BMOV	标准 RAM	4.1 μs	5.6 μs	56.0 μs	58.6 μs	526.0 μs	528.6 μs
		SRAM 卡	4.5 μs	6.1 μs	160.0 μs	162.6 μs	1571.0 μs	1573.6 μs
		Flash 卡 *1	4.3 μs	6.2 μs	255.0 μs	257.6 μs	2521.0 μs	2523.6 μs

*1 : Flash 卡中存储了文件寄存器的情况下，内部软元件→文件寄存器的模式将变为无处理。
*2: 在 Q00UCPU、Q01UCPU 中，不能使用存储卡。

(2) 文件寄存器→文件寄存器的情况下。

CPU	指令	文件寄存器存储对象存储器	1 字		1000 字		10000 字	
			最小	最大	最小	最大	最小	最大
QnHCPU	RBMOV	标准 RAM	20.0 μs		91.0 μs		775.0 μs	
		SRAM 卡	22.5 μs		545.0 μs		5300.0 μs	
	BMOV	标准 RAM	7.5 μs		77.0 μs		720.0 μs	
		SRAM 卡	8.5 μs		692.0 μs		7050.0 μs	
QnCPU	RBMOV	标准 RAM	45.5 μs		215.0 μs		1850.0 μs	
		SRAM 卡	50.0 μs		870.0 μs		8350.0 μs	
	BMOV	标准 RAM	17.5 μs		179.0 μs		1700.0 μs	
		SRAM 卡	18.5 μs		839.0 μs		8600.0 μs	
Q00UCPU Q01UCPU	RBMOV	标准 RAM	12.6 μs	35.3 μs	232.5 μs	256.1 μs	2211.5 μs	2235.1 μs
		SRAM 卡 *1	–	–	–	–	–	–
	BMOV	标准 RAM	7.7 μs	14.2 μs	227.5 μs	234.2 μs	2206.5 μs	2214.2 μs
		SRAM 卡 *1	–	–	–	–	–	–
Q02UCPU	RBMOV	标准 RAM	9.6 μs	31.5 μs	228.5 μs	252.3 μs	2208.5 μs	2231.3 μs
		SRAM 卡	9.6 μs	31.5 μs	378.5 μs	401.3 μs	3708.5 μs	3731.3 μs
	BMOV	标准 RAM	5.2 μs	11.8 μs	224.5 μs	232.3 μs	2204.5 μs	2212.3 μs
		SRAM 卡	5.2 μs	11.8 μs	374.5 μs	382.3 μs	3704.5 μs	3712.3 μs
Q03UD(E)CPU	RBMOV	标准 RAM	11.2 μs	16.7 μs	230.7 μs	237.1 μs	2210.7 μs	2217.1 μs
		SRAM 卡	11.6 μs	16.7 μs	380.7 μs	387.1 μs	3710.7 μs	3717.1 μs
	BMOV	标准 RAM	4.9 μs	6.7 μs	224.7 μs	227.1 μs	2204.7 μs	2207.1 μs
		SRAM 卡	5.2 μs	6.7 μs	374.7 μs	377.1 μs	3704.7 μs	3707.1 μs
Q04UD(E)HCPU Q06UD(E)HCPU Q10UD(E)HCPU Q13UD(E)HCPU Q20UD(E)HCPU Q26UD(E)HCPU	RBMOV	标准 RAM	9.3 μs	15.5 μs	118.0 μs	124.6 μs	1102.0 μs	1107.6 μs
		SRAM 卡	9.7 μs	15.5 μs	365.0 μs	371.6 μs	3571.0 μs	3578.6 μs
	BMOV	标准 RAM	4.3 μs	6.2 μs	113.0 μs	115.6 μs	1096.0 μs	1098.6 μs
		SRAM 卡	4.5 μs	6.1 μs	360.0 μs	362.6 μs	3566.0 μs	3568.6 μs

*1 ： 在 Q00UCPU、Q01UCPU 中不能使用存储卡。

7.18.20 用户信息指令

UMSG

Basic High performance Universal L CPU

UMSG

结构体梯形图

ST

```
ENO:= UMSG (EN, s);
```

中放入下述指令。

UMSG

输入自变量， EN: 执行条件 : 位

s: 显示模块中显示的字符串或存储显示的字符串的软元件的起始编号 : 字符串

输出自变量， ENO: 执行结果 : 位

设置数据	内部软元件		R, ZR	直接指令	J□\□		U□\G□	Zn	常数		其它数据
	位	字			位	字			K, H	实数字符串	
ⓢ	J		○	○						△*1	-

*1 : 只能使用字符串。

功 能

(1) 将ⓢ中指定的字符串数据，作为用户信息显示到显示模块。

显示ⓢ中直接指定的字符串（用 “ ”（双引号）围住进行指定），或ⓢ中指定的软元件编号开始至存储了 00H 的软元件编号为止的字符串。

	b15 ～ b8	b7 ～ b0
ⓢ	第2个字符	第1个字符
ⓢ+1	第4个字符	第3个字符
ⓢ+2	第6个字符	第5个字符
ⓢ+3	第8个字符	第7个字符
ⓢ+4	第10个字符	第9个字符
		00H

表示字符串的最后。

执行UMSG指令

(2) 在显示模块中可显示的字符串最多为半角的 128 个字符（全角的 64 个字符）。

要 点

使用字符串传送指令（\MOV 指令）将字符串存储到软元件中时，高位字节与低位字节将相反。
关于 \MOV 指令的有关内容，请参阅 6.4.4 项。

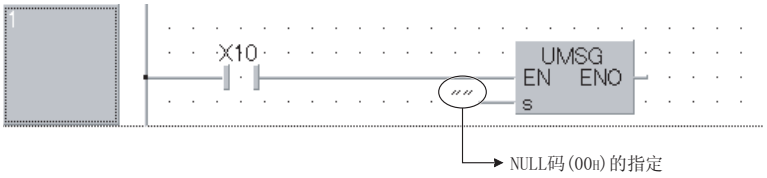
(3) 在 UMSG 指令的上升沿时显示用户信息。
在指令为 ON 的状态下，如果对字符串进行了变更，显示模块中将显示变更后的用户信息。

(4) 通过 UMSG 指令指定的字符串的显示是在 END 处理时进行。执行了多个 UMSG 指令的情况下，在 END 之前执行的 UMSG 指令将有效。执行了多个程序时，最后执行的 UMSG 指令将有效。

(5) 在未安装显示模块的状态下如果执行了指令，将变为无处理。

(6) 在用户信息的显示过程中，如果按下显示模块的 “ESC”，显示中的信息将消失。
若要再次显示信息，应通过显示模块的菜单画面执行 “用户信息”。

(7) 在指令的自变量中指定了 NULL 码 (00H) 的情况下，信息显示中时显示的信息将消失。
在指令的自变量中指定 NULL 码 (00H) 的方法如下所示。



关于显示模块的详细内容，请参阅 MELSEC-L CPU 模块用户手册（功能解说 / 程序基础篇）。

出 错

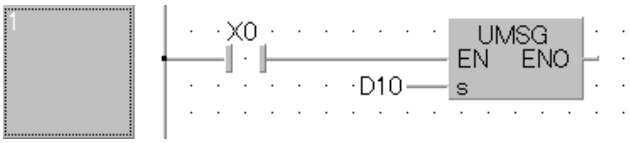
在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

- Ⓢ中指定的软元件编号以后，相应软元件的范围内不存在 NULL 码 (00H) 时。
（出错代码： 4101）
- Ⓢ的字符串的字符数超过了半角 128 个字符（全角 64 个字符）时。
（出错代码： 4100）

程序示例

(1) 以下为将 X10 置为 ON 时，将 D10 以后存储的字符串显示到显示模块中的程序。

[结构体梯形图]



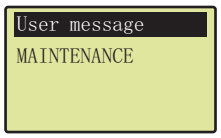
[ST]

```
UMSG(X0,D10);
```

[动作]

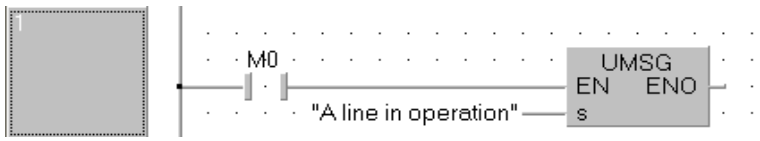
	B15 ～ b8	B7 ～ b0
D10	41H (A)	4DH (M)
D11	4EH (N)	49H (I)
D12	45H (E)	54H (T)
D13	41H (A)	4EH (N)
D14	43H (C)	4EH (N)
D15	20H(space)	45H (E)
D16	0000H	

执行UMSG指令



(2) 以下为将 M0 置为 ON 时，在显示模块中显示 “A 生产线运行中” 的程序。

[结构体梯形图]



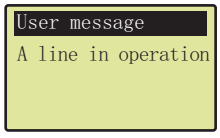
[ST]

```
UMSG(M0,"A 生产线运行中 ");
```

[动作]

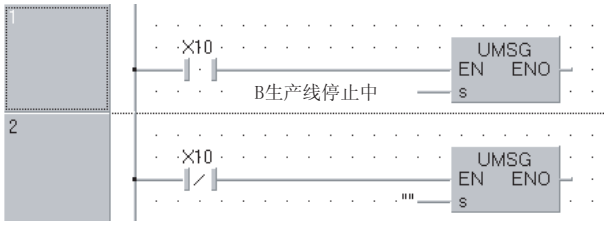
B15 ～ b8	B7 ～ b0
20H(space)	41H (A)
69H (i)	6CH (I)
65H (e)	6EH (n)
69H (i)	20H(space)
20H(space)	6EH (n)
70H (p)	6FH (o)
72H (r)	65H (e)
74H (t)	61H (a)
6FH (o)	69H (i)
20H(space)	6EH (n)
0000H	

执行UMSG指令



(3) 以下为将 X10 置为 ON 时，在显示模块显示 “B 生产线停止中”，将 X10 置为 OFF 时使信息消失的程序。

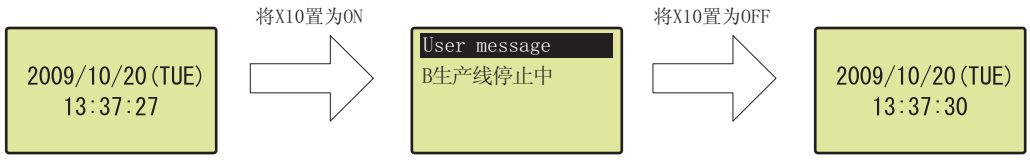
[结构体梯形图]



[ST]

```
UMSG(X10, "B 生产线停止中 ");
UMSG(NOT(X10), "");
```

[动作]



8 数据链接用指令

8.1 网络刷新指令

8.1.1 对指定模块的刷新

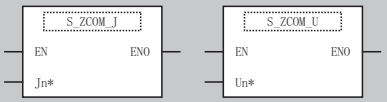
ZCOM

Basic | High performance | Universal | L CPU

S(P)_ZCOM_J(U)

P: 执行条件 :

结构体梯形图



ST

```
ENO:= S_ZCOM_J (EN, Jn*);
ENO:= S_ZCOM_U (EN, Un*);
```

中放入下述指令。

S_ZCOM_J	S_ZCOM_U
SP_ZCOM_J	SP_ZCOM_U

输入自变量，	EN:	执行条件	: 位
	Jn*:	自站的网络 No.（仅 QCPU(Q 模式)）	: ANY16
	Un*:	自站的网络模块的起始输入输出编号 (00 ～ FE: 将输入输出编号以 3 位数表示时的高 2 位)	: ANY16
输出自变量，	ENO:	执行结果	: 位

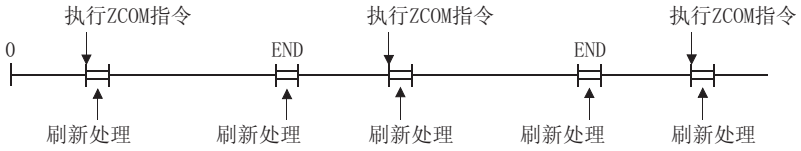
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
-	-								

功 能

ZCOM 指令用于在顺控程序的执行过程中在任意的时机进行刷新。
通过 ZCOM 指令进行刷新的对象如下所示。

- CC-Link IE 控制网络的刷新（刷新参数设置时）(仅 QCPU(Q 模式))
- MELSECNET/H 的刷新（刷新参数设置时）(仅 QCPU(Q 模式))
- CC-Link 的自动刷新（刷新软元件设置时）
- 智能功能模块的自动刷新（自动刷新设置时）

(1) 执行 ZCOM 指令时，CPU 模块暂时中断顺控程序的处理，进行 Jn/Un 中指定的网络模块的刷新处理。(LCPU 的情况下，不能通过 Jn 进行指定。)



(2) 在 ZCOM 指令中不进行下述处理。

(a) CPU 模块与编程工具的通信处理

(b) 其它站监视处理

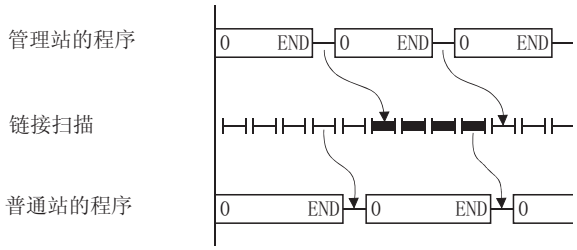
(c) 通过串行通信模块对其它智能功能模块的缓冲存储器的读取处理

(d) MELSECNET/H 的低速循环传送数据

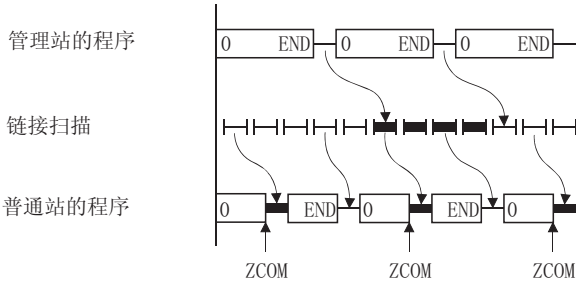
(3) 可编程控制器网络 *1 的情况下（仅 QCPU(Q 模式)）

(a) 自站的顺控程序的扫描时间长于其它站的扫描时间的情况下，为了确保从其它站获取数据而使用 ZCOM 指令。

1) 未使用 ZCOM 指令时的数据的发送接收示例



2) 使用了 ZCOM 指令时的数据的发送接收示例

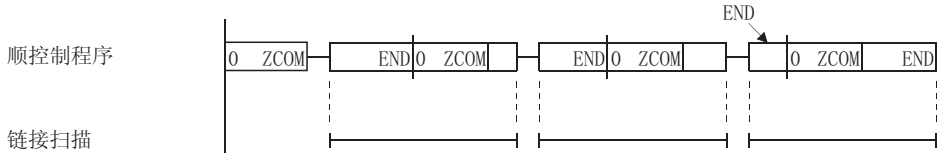


关于程控制器网络中的传送延迟时间的详细内容，请参阅下述手册。

• CC-Link IE 控制网络参考手册

• Q 系列 MELSECNET/H 网络系统参考手册

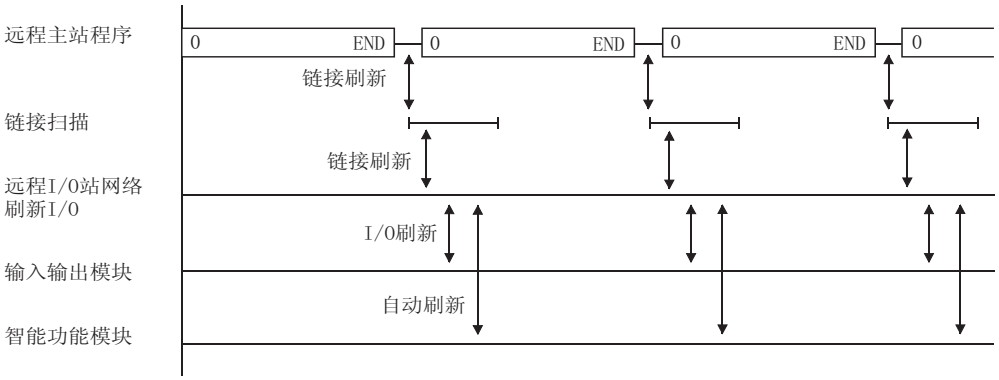
(b) 顺控程序的扫描时间短于链接扫描时间的情况下，即使使用 ZCOM 指令也不能加快数据的发送接收。



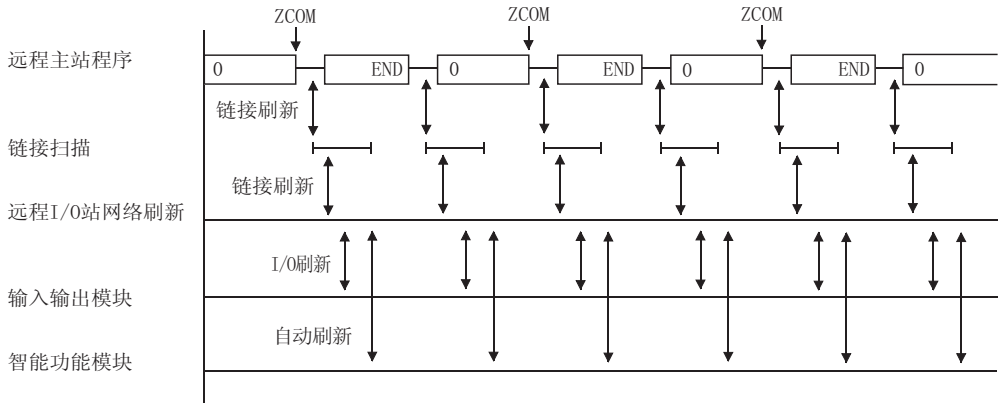
(4) 远程 I/O 网络的情况下（仅 QCPU(Q 模式)）

远程主站的链接刷新是在 CPU 模块的 “END 处理”中进行。
由于链接刷新结束时进行链接扫描，因此链接扫描与 CPU 模块的程序 “同步”。
在远程主站中使用 ZCOM 指令时，在 ZCOM 指令执行的时点进行链接刷新，链接刷新结束时进行链接扫描。
因此，在远程主站中使用 ZCOM 指令时，可以加快与远程 I/O 站的发送接收处理。

1) 未使用 ZCOM 指令的情况下



2) 使用了 ZCOM 指令的情况下



关于远程 I/O 网络的传送延迟时间的详细内容，请参阅下述手册。

• Q 系列 MELSECNET/H 网络系统参考手册（远程 I/O 网络篇）

(5) ZCOM 指令在顺控程序中的使用次数无限制。

但是顺控程序的扫描时间将有相当于刷新时间的延长，应加以注意。

(6) 通过在自变量中指定 Un，不仅网络模块可作为指定对象，也可将智能功能模块作为指定对象。

在这种情况下，进行智能功能模块的缓冲存储器的自动刷新。（替代 FROM/TO 指令。）

要 点

在恒定周期执行类型程序、中断程序中不能使用 ZCOM 指令。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

- 指定的网络 No. 未与自站相连接时。(QCPU(Q 模式）时）（出错代码 :4102)
- 起始输入输出编号中指定的模块不是网络模块 / 智能功能模块时。
  （基本型 QCPU、高性能型 QCPU 时）（出错代码 :2111)
- 起始输入输出编号中指定的模块不是网络模块 / 智能功能模块时。
  （通用型 QCPU、LCPU 时）（出错代码 :4102)

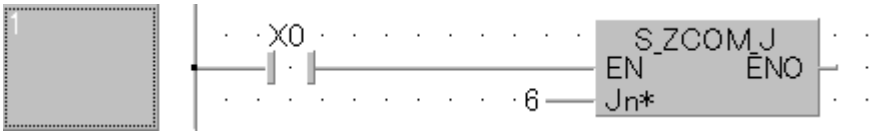
要 点

只希望与外围设备进行通信的情况下，应使用 COM 指令（7.6.5 项、7.6.6 项）。

程序示例

(1) 以下为 X0 处于 ON 状态下，进行网络 No.6 的网络模块的链接刷新的程序。

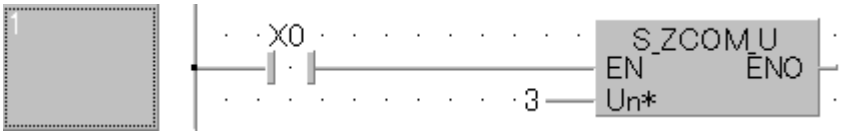
[结构体梯形图]




```
[ST]
IF(X0=TRUE) THEN
        S_ZCOM_J(TRUE,6);
END_IF;
```

(2) 以下为 X0 处于 ON 状态下，进行安装在起始输入输出编号 X/Y30 ～ X/Y4F 位置上的网络模块的链接刷新的程序。

[结构体梯形图]




```
[ST]
IF(X0=TRUE) THEN
        S_ZCOM_U(TRUE,3);
END_IF;
```

8.2 路由信息的读取 / 登录

8.2.1 路由信息的读取

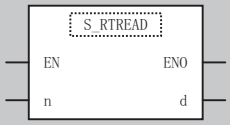
RTREAD

Basic | High performance | Universal | L CPU

S(P)_RTREAD

P: 执行条件 :

结构体梯形图



ST

ENO:= S_RTREAD (EN, n, d);

中放入下述指令。

S_RTREAD SP_RTREAD

输入自变量， EN: 执行条件 : 位
n: 传送目标网络 No.(1 ～ 239) : ANY16
输出自变量， ENO: 执行结果 : 位
d: 存储读取的数据的软元件的起始编号 : 位的数组 (0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
n	○	○		-				○	-
ⓓ	-	○		-				-	-

功 能

(1) 根据路由参数中设置的路由信息，对 n 中指定的传送目标网络 No. 的数据进行读取，存储到ⓓ以后。

(2) 在路由参数中未对 n 中指定的传送目标网络 No. 的数据进行设置的情况下，在ⓓ以后将存储 0。

(3) ⓓ以后中存储的数据内容如下所示。

(各数据的范围)

ⓓ[0]	中继目标网络No.	(1～239)
ⓓ[1]	中继目标站号	(1～120)
ⓓ[2]	虚拟	

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

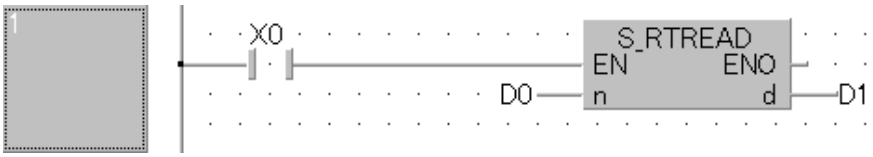
(1) n 的数据超出了 1 ～ 239 的范围时。 (出错代码 :4100)

(2) ⓓ中指定的软元件超出了相应软元件的范围时。(通用型 QCPU 时)
(出错代码 :4101)

程序示例

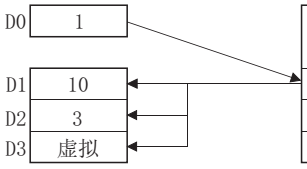
(1) 以下为 X0 变为 ON 时，对 D0 中指定的网络 No. 的路由信息进行读取的程序。

[结构体梯形图]




```
[ST]
IF(X0=TRUE) THEN
        S_RTREAD(TRUE,D0,D1);
END_IF;
```

[动作]



[路由参数的设置内容]

传送目标网络No.	中继目标网络No.	中继目标站号
1	10	3
2	10	2
3	10	1

8.2.2 路由信息的登录

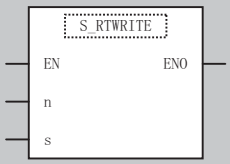
RTWRITE

Basic | High performance | Universal | L CPU

S(P)_RTWRITE

P: 执行条件 :

结构体梯形图



ST

```
ENO:= S_RTWRITE (EN, n, s);
```

中放入下述指令。

S_RTWRITE SP_RTWRITE

输入自变量， EN: 执行条件 : 位
n: 传送目标网络 No.(1 ～ 239) : ANY16
输出自变量， ENO: 执行结果 : 位
s: 存储写入数据的软元件的起始编号 : 位的数组 0..2)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
n	○	○		-				○	-
ⓢ	-	○		-				-	-

功 能

(1) 在路由参数的 n 中指定的传送目标网络 No. 的区域中，对ⓢ以后的路由数据进行登录。

(2) ⓢ以后中存储的数据的内容如下所示。

(各数据的范围)

ⓢ[0]	中继目标网络No.	(0～239)
ⓢ[1]	中继目标站号	(0～120)
ⓢ[2]	虚拟	

(3) n 中指定的传送目标网络 No. 的数据未在路由参数中进行设置的情况下，将变更为ⓢ以后的数据。

(4) ⓢ以后的数据（ⓢ[0]～ⓢ[2]）全部为 0 的情况下，将 n 中指定的传送目标网络 No. 的数据从路由参数中删除。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) n 的数据超出了 1 ～ 239 的范围时。 (出错代码 :4100)

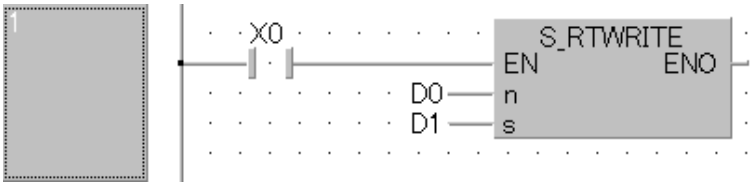
(1) ⓢ以后的数据超出了各设置范围时。 (出错代码 :4100)
网络参数的路由参数中登录的路由信息的登录数与 RTWRITE 指令中登录的路由信息的登录数的合计超过了 64 时。
(出错代码 :4100)

(1) ⓢ中指定的软元件超出了相应软元件的范围时。(通用型 QCPU 时)
(出错代码 :4101)

程序示例

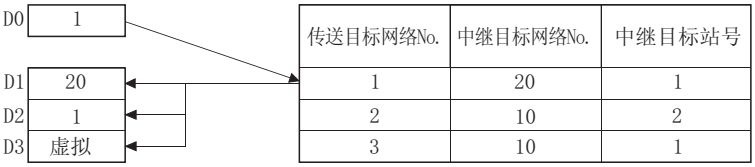
(1) 以下为 X0 变为 ON 时，将 D1 ～ D3 中指定的路由信息写入到 D0 中指定的网络 No. 的网络模块中的程序。

[结构体梯形图]




```
[ST]
IF(X0=TRUE) THEN
        S_RTWRITE(TRUE,D0,D1);
END_IF;
```

[动作]　[路由参数的设置内容]



传送目标网络No.	中继目标网络No.	中继目标站号
1	20	1
2	10	2
3	10	1

9 多 CPU 间专用

9.1 至本站 CPU 共享存储器的写入

在多 CPU 系统中通过 S(P)_TO 指令或 TO(P) 指令对本站的 CPU 共享存储器进行写入。

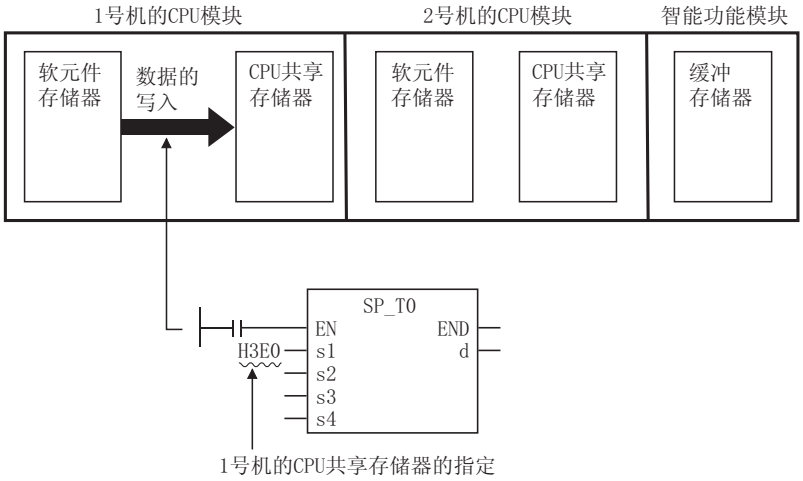
S(P)_TO 指令与 TO(P) 指令的使用可否如下表所示。

CPU 模块型号		S(P)_TO 指令	TO(P) 指令
基本型 QCPU	Q00J	禁止使用	禁止使用
	Q00 Q01	可以使用	可以使用
高性能型 QCPU	Q02(H) Q06H Q12H Q25H	可以使用	禁止使用
通用型 QCPU	Q00UJ	禁止使用	禁止使用
	Q00U Q01U Q02U Q03UD Q03UDE Q04UDH Q04UDEH Q06UDH Q06UDEH Q10UDH Q10UDEH Q13UDH Q13UDEH Q20UDH Q20UDEH Q26UDH Q26UDEH	可以使用	可以使用
LCPU	L02 L26-BT	禁止使用	禁止使用

(1) S(P)_TO 指令的动作

通过使用 S(P)_TO 指令，可以将数据写入到本站 CPU 模块的 CPU 共享存储器中。

在 1 号机的 CPU 模块中，执行了 S(P)_TO 指令时的处理如下图所示。



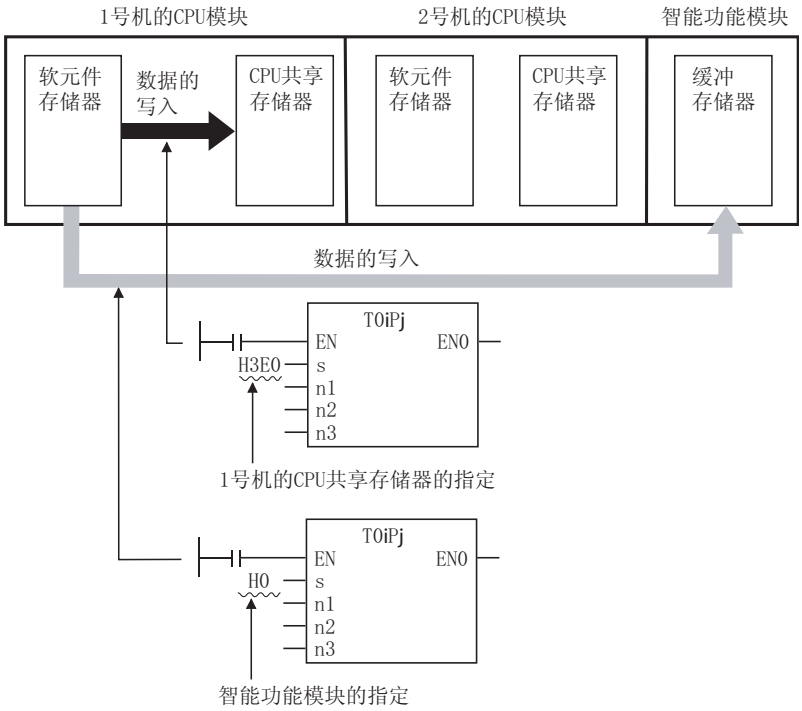
(2) TO(P) 指令的动作

通过使用 TO(P) 指令，可以将软元件存储器的数据写入到下述存储器中。

• 本站 CPU 模块的 CPU 共享存储器

• 智能功能模块的缓冲存储器

在 1 号机的 CPU 模块中，执行了 TO(P) 指令时的处理如下图所示。



要 点

对于基本型 QCPU(Q00CPU、Q01CPU) 与通用型 QCPU，通过 S(P)_TO 指令或 TO(P) 指令均可对 CPU 共享存储器进行写入，但在对本站的 CPU 共享存储器进行写入的情况下，使用 TO(P) 指令时可缩短处理时间，因此建议使用 TO(P) 指令。

备 注

通过 TO 指令对智能功能模块的缓冲存储器进行写入时，请参阅 7.8.2 项。

9.1.1 至本站 CPU 共享存储器的写入

S_TO

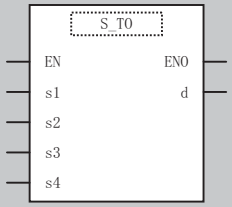
Ver. Basic　Ver. High performance　Universal　L CPU

Q00CPU/Q01CPU：序列号的前 5 位数为 “04122” 以后
高性能型 QCPU：功能版本 B 以后

S(P)_TO

P：执行条件　　：

结构体梯形图



ST

ENO:= S_TO (EN, s1, s2, s3, s4, d);

中放入下述指令。

S_TO　　SP_TO

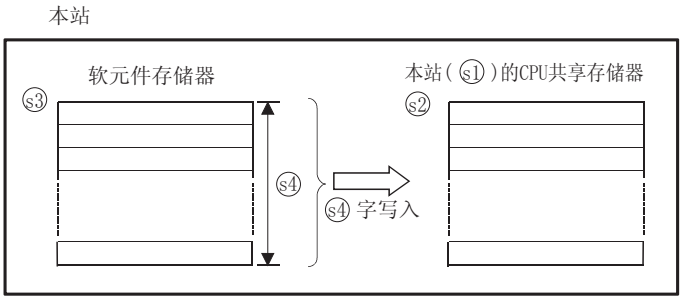
输入自变量，EN:　执行条件　：位
s1:　本站的起始输入输出编号　：ANY16
s2:　写入目标的本站的 CPU 共享存储器地址（BIN16 位）　：ANY16
基本型 QCPU：0 ～ 511
高性能型 QCPU、通用型 QCPU：0 ～ 4095
s3:　存储写入数据的软元件的起始编号　：ANY16
s4:　写入点数　：ANY16
基本型 QCPU：1 ～ 320
高性能型 QCPU：1 ～ 256
通用型 QCPU：1 ～ 2048
输出自变量，ENO:　执行结果　：位
d:　写入结束时置为 1 个扫描 ON 的本站软元件　：位

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据
	位	字		位	字				
⑤1	–	○		–				○	–
⑤2	–	○		–				○	–
⑤3	–	○		–				–	–
⑤4	–	○		–				○	–
ⓓ	○	○		–				–	–

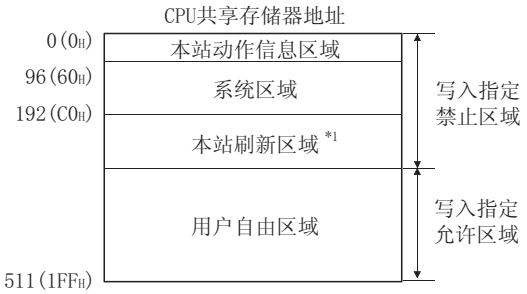
功 能

(1) 从本站 CPU 模块的③ 中，将④ 字的软元件数据，写入到本站 CPU 模块的② 中指定的 CPU 共享存储器地址以后。

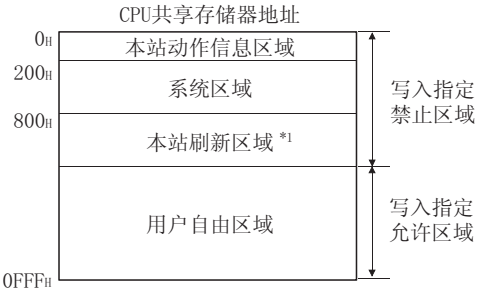
写入结束时，ⓓ 中指定的结束位将变为 ON。



(a) 基本型 QCPU 时的 CPU 共享存储器地址



(b) 高性能型 QCPU、通用型 QCPU*2 时的 CPU 共享存储器地址



*1 ： 未进行自动刷新设置的情况下，可以作为用户自由区域使用。
此外，即使在进行了自动刷新设置的情况下，自动刷新发送范围以后也可以作为用户自由区域使用。

*2 ： 在 S(P)_TO 指令中，不能对通用型 QCPU 的多 CPU 间高速通信区域进行写入。

(2) 写入点数为 0 时，变为无处理，结束软元件也不变为 ON。

(3) 在 1 个号机的 1 个扫描中只能执行 1 次 S(P)_TO 指令。
在有 2 个位置以上的执行条件同时成立的情况下，将自动进行握手，后执行的 S(P)_TO 指令将不执行处理。

(4) 写入数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 320
高性能型 QCPU	1 ～ 256
通用型 QCPU	1 ～ 2048

要 点

使用智能功能模块软元件也可以对 CPU 共享存储器进行数据写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将发生运算出错，出错标志（SM0）将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出了下述范围时。（出错代码 :4101）

- 写入点数 ④ 超出了设置数据的指定范围时。
- 写入目标的本站的 CPU 共享存储器地址 ② 的起始超出了 CPU 共享存储器地址的范围时。
- 写入目标的本站的 CPU 共享存储器地址 ② + 写入点数 ④ 超出了 CPU 共享存储器地址的范围时。
- 存储写入数据的软元件的起始编号 ③ + 写入点数 ④ 超出了软元件范围时。

(2) 入目标的本站的 CPU 共享存储器地址 ② 中，指定了本站动作信息区域、系统区域或本站刷新区域时。

（高性能型 QCPU）（出错代码 :4101）
（基本型 QCPU、通用型 QCPU）（出错代码 :4111）

(3) 本站的起始输入输出编号 ① 中指定了除本站以外时。

（高性能型 QCPU）（出错代码 :2107）
（基本型 QCPU、通用型 QCPU）（出错代码 :4112）

(4) CPU 模块的起始输入输出编号中指定的位置处 CPU 模块不存在时。（出错代码 :2110）

(5) 在本站的起始输入输出编号 ① 中，指定了除 3E0H/3E1H/3E2H/3E3H 以外时。（出错代码 :4100）

(6) 指定的指令不正确时。（出错代码 :4002）

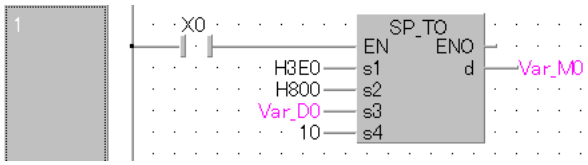
(7) 指定的软元件数有错误时。（出错代码 :4003）

(8) 指定了不能使用的软元件时。（出错代码 :4004）

程序示例

以下为 X0 变为 ON 时，将 Var_D0 算起的 10 点的数据存储到 1 号机的 CPU 共享存储器的 800H 的地址中的程序。

[结构体梯形图]



[ST]

```
SP_TO(X0,H3E0,H800,Var_D0,10,Var_M0);
```

备 注

对于 ①，以将安装了 CPU 模块的插槽的起始输入输出编号以 16 进制数 4 位表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始输入输出编号	3E00	3E10	3E20	3E30
①	3E0	3E1	3E2	3E3

9.1.2 至本站 CPU 共享存储器的写入

TO, DTO

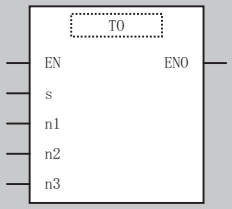
Ver. Basic | High performance | Universal | L CPU

Q00CPU/Q01CPU: 序列号的前 5 位数为 “04122” 以后

TO(P)
DTO(P)

P: 执行条件　　: ↓

结构体梯形图



ST

ENO:= TO (EN, s, n1, n2, n3);

中放入下述指令。

TO	TOP
DTO	DTOP

输入自变量， EN:　执行条件　：位

s:　写入数据或存储写入数据的软元件的起始编号　：ANY16

n1:　本站的起始输入输出编号　：ANY16
基本型 QCPU: 3E0H
通用型 QCPU: 3E0H ～ 3E3H

n2:　写入目标的本站的 CPU 共享存储器地址　：ANY16
基本型 QCPU: 192 ～ 511
通用型 QCPU: 2048 ～ 4095
10000 ～ 24335*1

n3:　写入点数 1 ～ 320　：ANY16
基本型 QCPU: 1 ～ 320
通用型 QCPU: 1 ～ 14336*1

输出自变量， ENO:　执行结果　：位

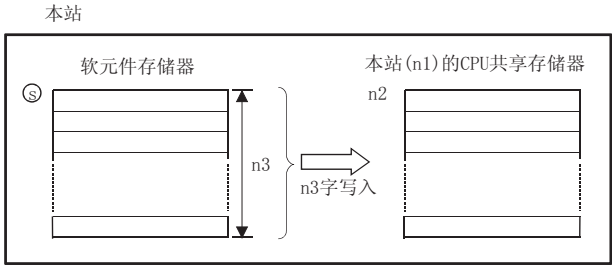
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据 U
	位	字		位	字				
ⓢ	○			-				○	-
n1	○			○				○	○
n2	○			○				○	-
n3	○			○				○	-

*1 ：设置范围取决于多 CPU 间高速通信功能的自动刷新设置范围。

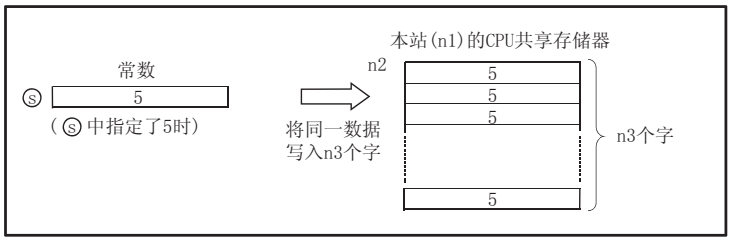
功 能

TO

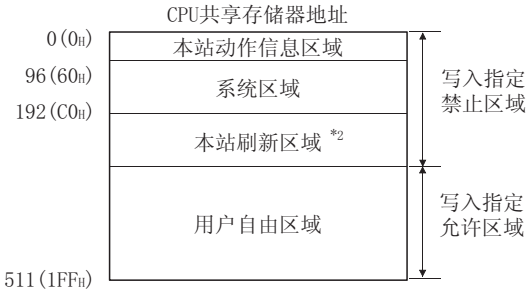
(1) 将本站 CPU 模块的Ⓢ算起的 n3 字的软元件数据，写入到本站 CPU 模块的 n2 中指定的 CPU 共享存储器地址以后。



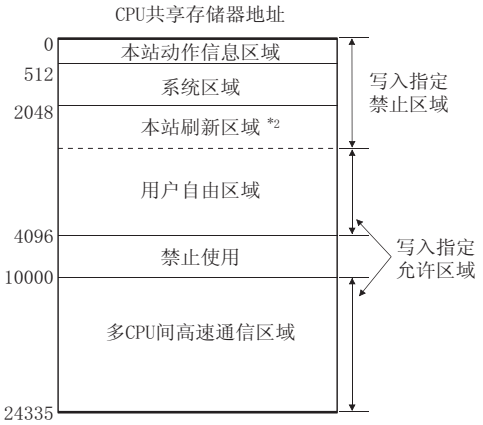
Ⓢ中指定了常数的情况下，将同一个数据（Ⓢ中指定的值），写入到指定的 CPU 共有存储器算起的 n3 字中。



(a) 基本型 QCPU 时的 CPU 共享存储器地址



(a) 通用型 QCPU 时的 CPU 共享存储器地址 *3



*2 ： 未进行自动刷新设置的情况下，可以作为用户自由区域使用。
此外，即使进行了自动刷新设置，自动刷新发送范围以后也可以作为用户自由区域使用。

*3 ： 在 Q02UCPU 中，不能对多 CPU 间高速通信区域进行写入。

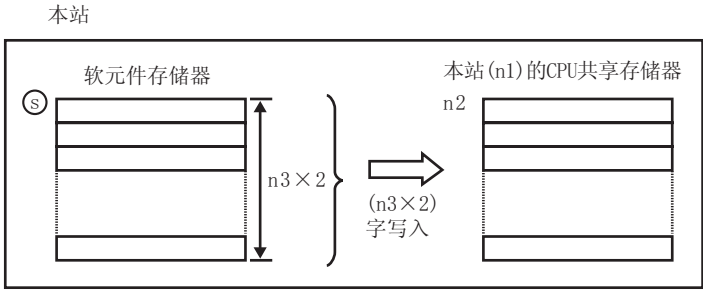
(2) 写入点数为 0 时进行无处理。

(3) 写入数据数根据对象 CPU 模块而有所不同。

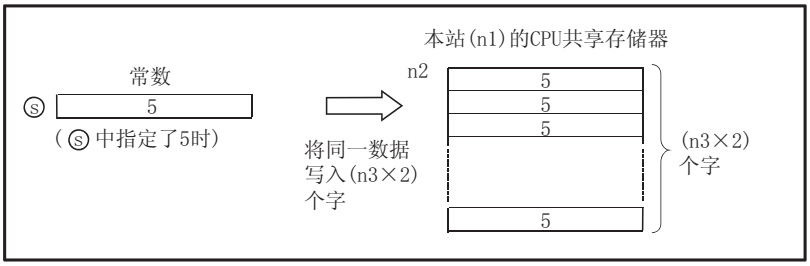
CPU 模块	写入点数
基本型 QCPU	1 ～ 320
通用型 QCPU	1 ～ 14336

DTO

(1) 将本站 CPU 模块的Ⓢ算起的 (n3 × 2) 字的软元件数据，写入到本站 CPU 模块的 n2 中指定的 CPU 共享存储器地址以后。



Ⓢ中指定了常数的情况下，将同一个数据（Ⓢ中指定的值），写入到指定的 CPU 共有存储器算起的 (n3 × 2) 字中。



(2) 写入点数为 0 时进行无处理。

(3) 写入数据数根据对象 CPU 模块而有所不同。

CPU 模块	写入点数
基本型 QCPU	1 ～ 160
通用型 QCPU	1 ～ 7168

要 点

使用智能功能模块软元件也可以对 CPU 共享存储器进行数据写入。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出以下范围时。 (出错代码 :4101)

- 写入点数 (n3) 超出设置数据的指定范围时。
- 写入目标的本站的 CPU 共享存储器地址 (n2)+ 写入点数 (n3) 超出了 CPU 共享存储器的范围时。
- 存储写入数据的起始软元件编号 (Ⓢ)+ 写入点数 (n3) 超出了软元件范围时。
- 写入目标的本站的 CPU 共享存储器地址 (n2) 的起始中，指定了跨越写入允许区域的值时。

(1) 写入目标的本站的 CPU 共享存储器地址 (n2) 的起始不是有效值时。 (出错代码 :4111)

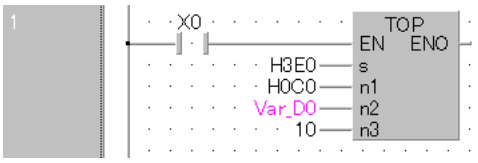
(1) (n1) 中指定了除本站以外时。(但是，指定了其它机号的多 CPU 间高速通信区域时除外。) (出错代码 :4112)

(1) CPU 模块的起始输入输出编号中指定的位置处 CPU 模块不存在时。 (出错代码 :2110)

程序示例

(1) 以下为 X0 变为 ON 时，将 Var_D0 算起的 10 点的数据，存储到 1 号机的 CPU 共享存储器的 10000 的地址以后的程序。

[结构体梯形图]

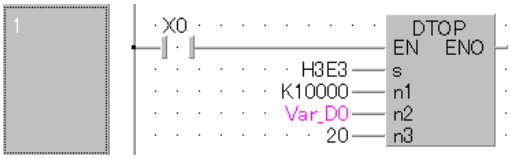


[ST]

```
TOP(X0,H3E0,K10000,Var_D0,10);
```

(2) 以下为 X0 变为 ON 时，将 Var_D0 算起的 20 点的数据，存储到 4 号机的 CPU 共享存储器的 10000 地址以后的程序。

[结构体梯形图]



[ST]

```
DTOP(X0,H3E3,K10000,Var_D0,20);
```

备 注

对于 n1，以将安装了 CPU 模块的插槽的起始输入输出编号以 16 进制数 4 位表示时的高 3 位进行指定。

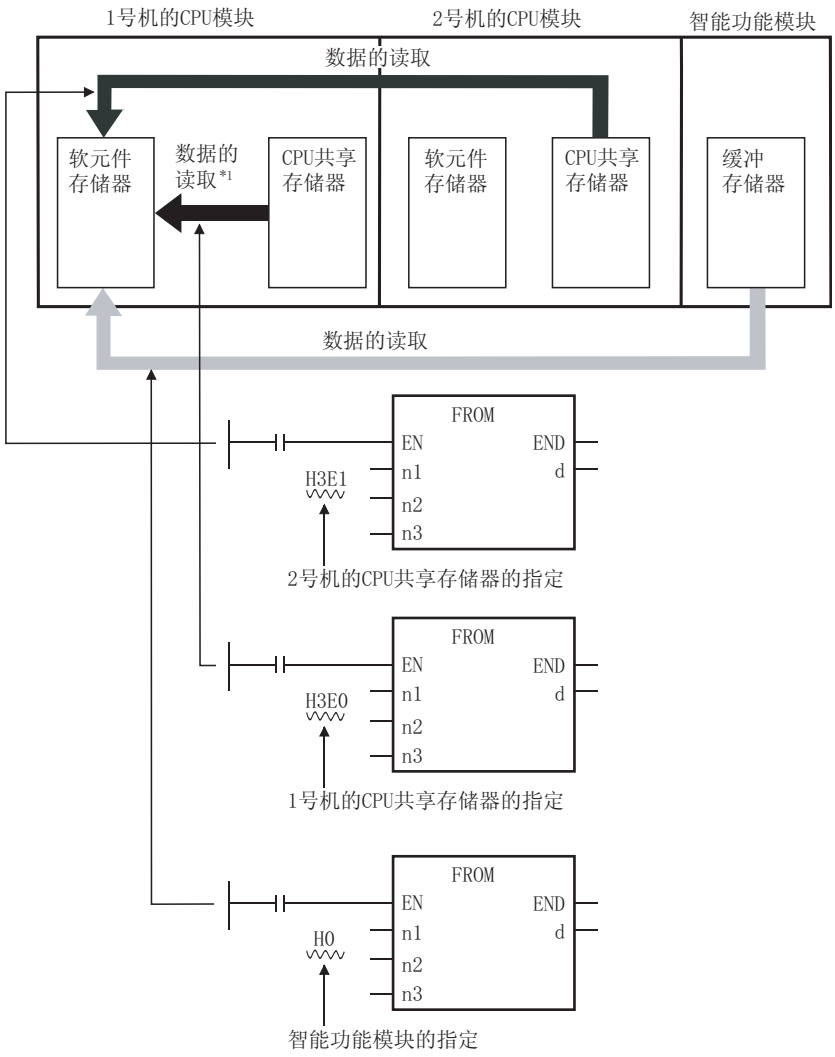
	CPU 插槽	插槽 0	插槽 1	插槽 2
起始输入输出编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

9.2 从其它机号 CPU 共享存储器中读取

通过使用多 CPU 系统的 FROM(P) 指令，可以从下述存储器中进行读取。

- 智能功能模块的缓冲存储器
- 其它机号 CPU 模块的 CPU 共享存储器
- 本站 CPU 模块的 CPU 共享存储器（在基本型 QCPU、通用型 QCPU 中可以执行）

在 1 号机中执行了 FROM(P) 指令时的处理如下图所示。



*1 ： 在基本型 QCPU、通用型 QCPU 中可以执行。

备 注

通过 FROM(P) 指令对智能功能模块的缓冲存储器进行读取时，请参阅 7.8.1 项。

9.2.1 从其它机号 CPU 共享存储器中读取

FROM, DFRO

Basic Ver. / High performance Ver. / Universal / L CPU (×)

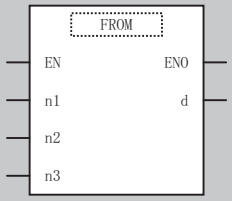
Q00CPU/Q01CPU: 序列号的前 5 位数为 “04122”以后
高性能型 QCPU: 功能版本 B 以后

1 使用基本型 QCPU、通用型 QCPU 时

FROM(P)
DFRO(P)

P: 执行条件 : ↰

结构体梯形图



ST

ENO:= FROM (EN, n1, n2, n3, d);

[ ] 中放入下述指令。

FROM	FROMP
DFRO	DFROP

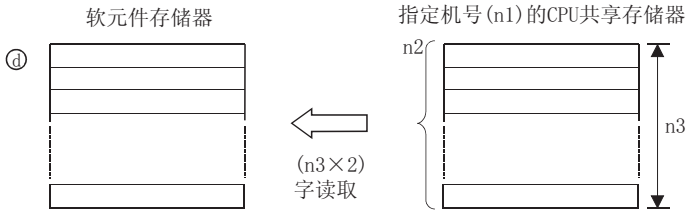
输入自变量, EN: 执行条件 : 位
n1: 读取对象的 CPU 模块的起始输入输出编号 : ANY16
基本型 QCPU:3E0H ～ 3E2H
通用型 QCPU:3E0H ～ 3E3H
n2: 读取数据的起始地址 : ANY16
基本型 QCPU:0 ～ 512
通用型 QCPU:0 ～ 4095, 10000 ～ 24335*1
n3: 读取数据数 : ANY16
基本型 QCPU:
FROM(P):1 ～ 512
DFRO(P):1 ～ 256
通用型 QCPU:
FROM(P):1 ～ 14336*1
DFRO(P):1 ～ 7168*1
输出自变量, ENO: 执行结果 : 位
d: 存储读取的数据的软元件起始编号 : ANY16

*1 : 设置范围取决于多 CPU 间高速通信功能的自动刷新设置范围。

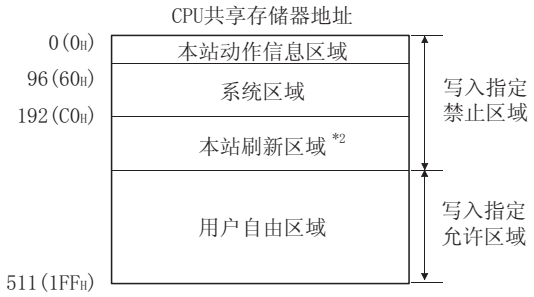
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据 U
	位	字		位	字				
n1	–	○		○				○	○
n2	–	○		○				○	–
n3	–	○		○				○	–
ⓓ	–	○		–				–	–

功 能

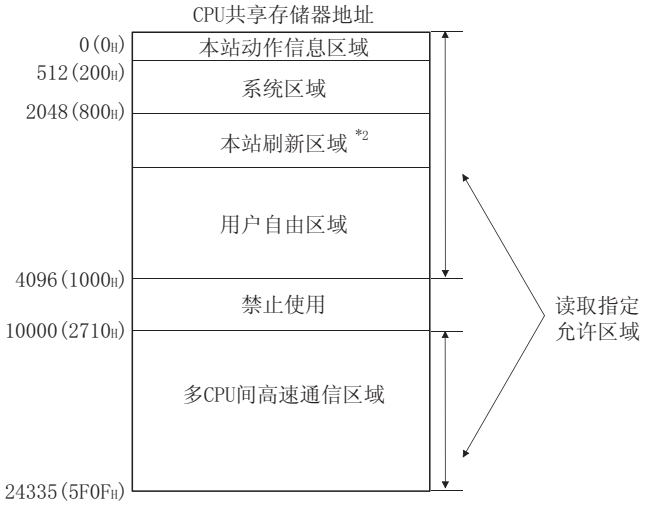
(1) 从 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址开始，读取 n3 字的数据后，存储到ⓓ中指定的软元件以后。



(a) 基本型 QCPU 时的 CPU 共享存储器地址



(b) 通用型 QCPU 时的 CPU 共享存储器地址 *3



*2 ： 未进行自动刷新设置的情况下，可以作为用户自由区域使用。
此外，即使进行了自动刷新设置的情况下，自动刷新发送范围以后也可以作为用户自由区域使用。

*3 ： 在 Q02UCPU 中不能对多 CPU 间高速通信区域进行读取。

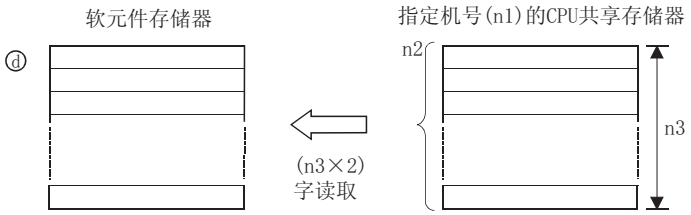
(2) 读取数据 n3 为 0 时，变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	读取点数
基本型 QCPU	1 ～ 512
通用型 QCPU	1 ～ 14336

DFRO

(1) 从 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址开始，读取 (n3 × 2) 字的数据后，存储到ⓓ中指定的软元件以后。



(2) 读取数据 n3 为 0 时，变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	读取点数
基本型 QCPU	1 ～ 256
通用型 QCPU	1 ～ 7168

要 点

使用智能功能模块软元件也可以对 CPU 共享存储器进行数据读取。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出以下范围时。 （出错代码 :4101）

- 进行读取的 CPU 共享存储器地址 n2 的起始超出了 CPU 共享存储器的范围时。
- 进行读取的 CPU 共享存储器地址 n2+ 读取点数 n3 超出了 CPU 共享存储器的范围时。
- 读取数据存储软元件编号ⓓ+ 读取点数 n3 超出了指定软元件范围时。

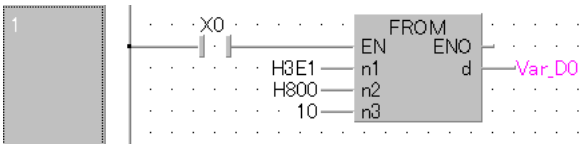
(1) CPU 模块的起始输入输出编号中指定的位置处 CPU 模块不存在时。 （出错代码 :2110）

(1) 进行读取的 CPU 共享存储器地址 (n2) 的起始不是有效值时。(4097 ～ 9999)
（出错代码 :4101）

程序示例

(1) 以下为 X0 变为 ON 时，从 2 号机的 CPU 共享存储器的 800H 地址开始将 10 点的数据存储到 Var_D0 以后的程序。

[结构体梯形图]

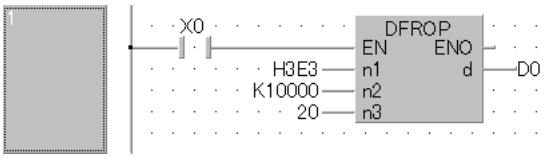


[ST]

```
FROM(X0,H3E1,H800,10,Var_D0);
```

(2) 以下为 X0 变为 ON 时，从 4 号机的 CPU 共享存储器的 10000 地址开始将 20 点的数据存储到 D0 以后的程序。

[结构体梯形图]



[ST]

```
DFROMP(X0,H3E3,K10000,20,D0);
```

备 注

对于 n1，以将安装了 CPU 模块的插槽的起始输入输出编号以 16 进制数 4 位表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始输入输出编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

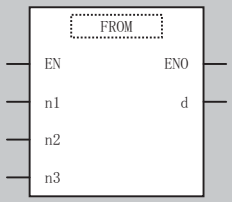
采用了 FROM/TO 指令的自动互锁。

2 使用高性能型 QCPU 时

FROM(P)

P: 执行条件 : ↓

结构体梯形图



ST

ENO:= FROM (EN, n1, n2, n3, d);

□ 中放入下述指令。

FROM FROMP

输入自变量， EN: 执行条件 : 位

n1: 读取对象的 CPU 模块的起始输入输出编号 : ANY16
3E0H ～ 3E3H

n2: 读取数据的起始地址 : ANY16
0 ～ 4095

n3: 读取数据数 : ANY16
1 ～ 4096

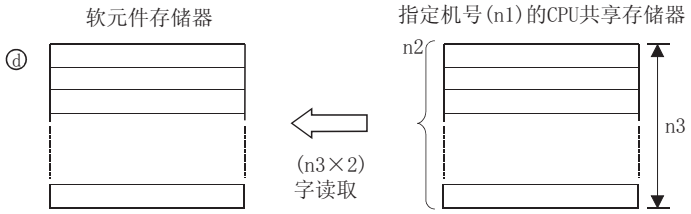
输出自变量， ENO: 执行结果 : 位

d: 存储读取的数据的软元件起始编号 : ANY16

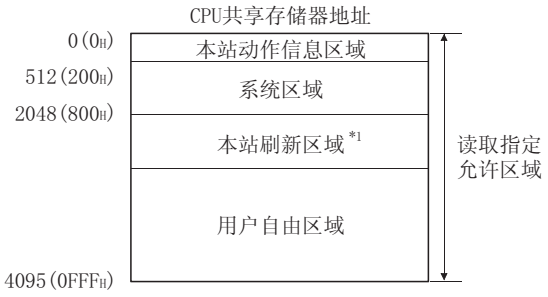
设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数 K,H	其它数据 U
	位	字		位	字				
n1	–	○		○				○	○
n2	–	○		○				○	–
n3	–	○		○				○	–
ⓓ	–	○		–				–	–

功 能

(1) 从 n1 中指定的 CPU 模块的 n2 中指定的 CPU 共享存储器地址开始，读取 n3 字的数据后，存储到ⓓ中指定的软元件以后。



高性能型 QCPU 时的 CPU 共享存储器地址



*1 ： 未进行自动刷新设置的情况下，可以作为用户自由区域使用。
此外，即使进行了自动刷新设置的情况下，自动刷新发送范围以后也可以作为用户自由区域使用。

(2) 读取数据 n3 为 0 时，变为无处理。

(3) 读取数据数根据对象 CPU 模块而有所不同。

CPU 模块	读取点数
高性能型 QCPU	1 ～ 4096

要 点

使用智能功能模块软元件也可以对 CPU 共享存储器进行数据读取。
关于智能功能模块软元件，请参阅所使用的 CPU 模块的用户手册（功能解说 / 程序基础篇）。

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的数据超出以下范围时。 (出错代码 :4101)

- 进行读取的 CPU 共享存储器地址 n2 的起始超出了 CPU 共享存储器的范围时。
- 进行读取的 CPU 共享存储器地址 n2+ 读取点数 n3 超出了 CPU 共享存储器的范围时。
- 读取数据存储软元件编号 ⓓ + 读取点数 n3 超出了指定软元件范围时。

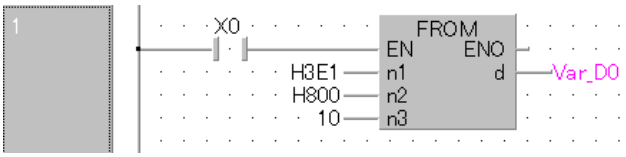
(1) CPU 模块的起始输入输出编号中指定的位置处 CPU 模块不存在时。 (出错代码 :2110)

(1) 进行读取的 CPU 共享存储器地址 (n2) 的起始不是有效值时。(4097 ～ 9999)
(出错代码 :4101)

程序示例

以下为 X0 变为 ON 时，从 2 号机的 CPU 共享存储器的 800H 地址开始将 10 点的数据存储到 Var_D0 以后的程序。

[结构体梯形图]



[ST]

```
FROM(X0,H3E1,H800,10,Var_D0);
```

备 注

对于 n1，以将安装了 CPU 模块的插槽的起始输入输出编号以 16 进制数 4 位表示时的高 3 位进行指定。

	CPU 插槽	插槽 0	插槽 1	插槽 2
起始输入输出编号	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

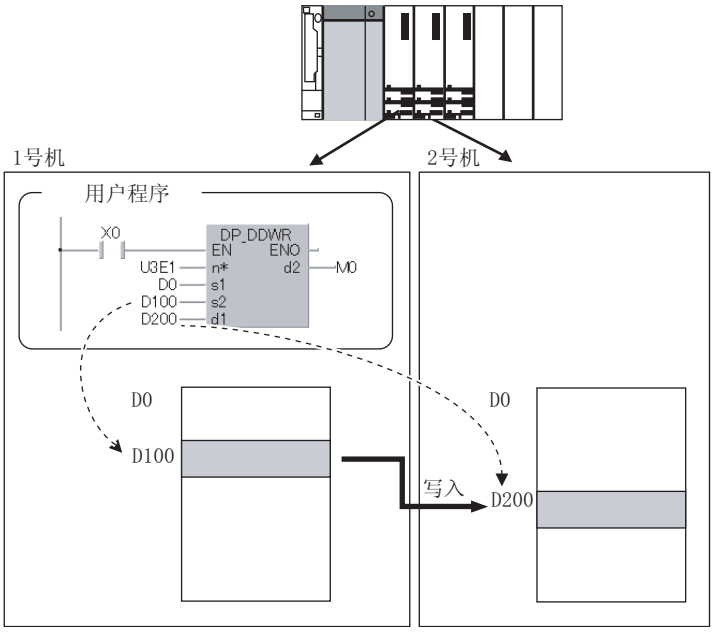
采用了 FROM/TO 指令的自动互锁。

10 多 CPU 间高速通信专用指令

10.1 概要

多 CPU 间高速通信专用指令是指，从通用型 QCPU 向其它机号的通用型 QCPU 进行软元件数据的写入 / 读取的指令。

通过多 CPU 间高速通信专用指令进行从 1 号机至 2 号机的写入时的动作如下图所示。



要 点

若要使用多 CPU 间高速通信专用指令，本站、其它机号（指令的执行对象机号）均只能使用下述 CPU 模块。

- 序列号的前 5 位数为 “10012” 以后的 Q03UDCPU、Q04UDHCPU、Q06UDHCPU
- Q10UDHCPU、Q13UDHCPU、Q26UDHCPU
- QnUDE(H)CPU

(1) 用于执行多 CPU 间高速通信专用指令的系统、参数设置

对于多 CPU 间高速通信专用指令功能，在下述系统配置、参数设置时可以执行。

- 1 号机中使用了 QnUD(H)CPU 或 QnUDE(H)CPU。
- 使用多 CPU 间高速通信主基板 (Q3 □ DB)。
- 在可编程控制器参数的多 CPU 设置中设置为 “使用多 CPU 间高速通信功能”。

(2) 可以写入 / 读取的软元件

可以写入 / 读取的软元件名

可以通过多 CPU 间高速通信专用指令与其它机号的通用型 QCPU 进行写入 / 读取的软元件如下表所示。

分类	类型	软元件名	对象软元件设置 允许 / 禁止	备注
内部用户软元件	位软元件	X,Y,M,L,B,F,SB	△	设置时的必要条件 • 必须为 16 位 (4 位数) 的位数指定。 • 开始位软元件必须是 16(10进) 的倍数。
	字软元件	T,ST,C,D,W,SW	○	—
内部系统软元件	位软元件	SM	△	设置时的必要条件 • 必须为 16 位 (4 位数) 的位数指定。 • 开始位软元件必须是 16(10进) 的倍数。
	字软元件	SD	○	—
文件寄存器	字软元件	R,ZR	○	—

○ : 可以设置 △ : 可以带条件设置

要 点

SB、SW、SM、SD 中包含有系统信息区域。

通过多 CPU 间高速通信专用指令的 D(P)_DDWR 指令对上述软元件进行写入的情况下，应注意不要破坏系统信息。

(3) 软元件的指定方法及可以写入 / 读取的软元件范围

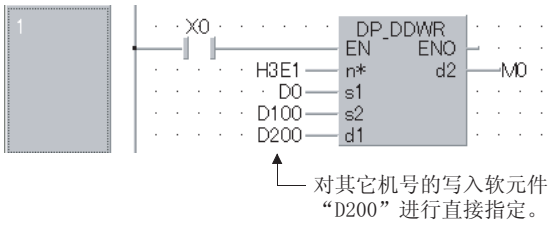
在其它机号 CPU 的软元件的指定方法中，有软元件指定及字符串指定这 2 种类型。

在软元件指定及字符串指定中，可对其它机号进行写入 / 读取的软元件范围有所不同。

(a) 软元件指定

软元件指定是指，对要写入 / 读取的其它机号的软元件进行直接指定的方法。

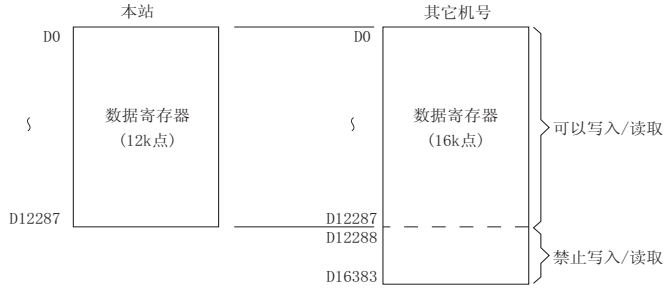
DP_DDWR 指令的软元件指定程序



在软元件指定中，可以在本站的软元件范围内进行写入 / 读取。

例如，本站的数据寄存器为 12k 点，其它机号的数据寄存器为 16k 点的情况下，可以从其它机号的数据寄存器的起始开始进行 12k 点的写入 / 读取。

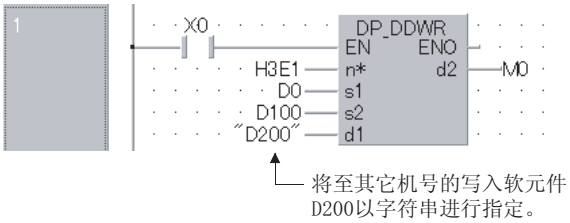
软元件指定时的写入 / 读取范围



(b) 字符串指定

字符串指定是指，将进行写入 / 读取的其它机号的软元件通过字符串进行指定的方法。

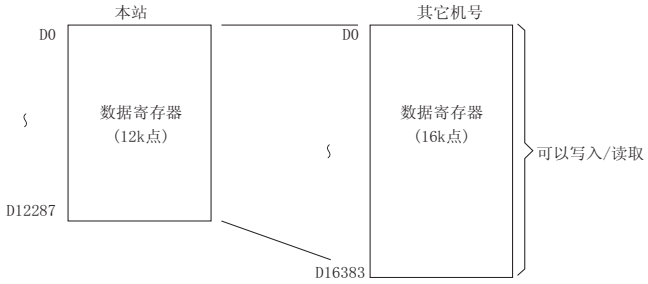
DP_DDWR 指令的字符串指定程序



在字符串指定中，可以对其它机号的软元件的所有范围进行写入 / 读取。

例如，本站的数据寄存器为 12k 点，其它机号的数据寄存器为 16k 点的情况下，可以从其它机号的数据寄存器的起始开始进行 16k 点的写入 / 读取。

字符串指定时的写入 / 读取范围



要 点

字符串指定的注意事项如下所示。

可进行字符串指定的最多允许字符数为 32 个字符。

软元件 No. 的高位中无论是否附加了 “0”，均作为同一软元件处理。例如 “D1” 与 “D0001” 均作为 D1 处理。

对于通过大写字母及小写字母指定的软元件，作为同一软元件处理。例如 “D1” 与 “d1” 均作为 D1 处理。

通过字符串指定了其它机号 CPU 中不存在的软元件的情况下，将变为指令异常结束状态。

(4) 多 CPU 间高速通信区域的管理

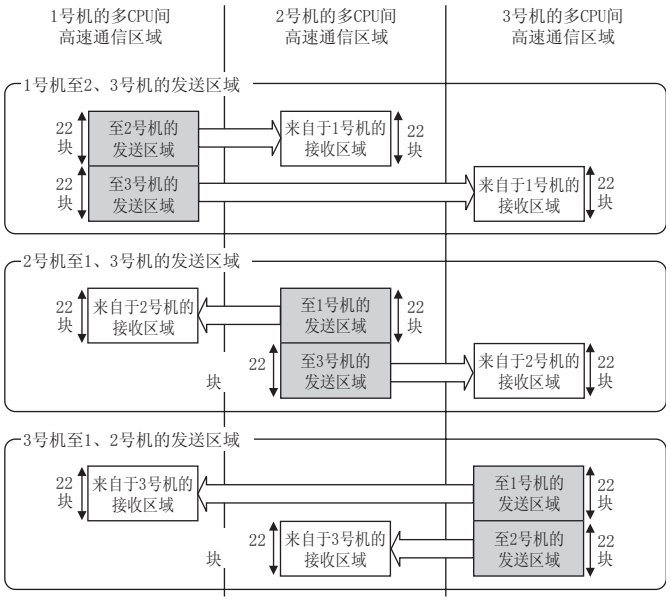
(a) 多 CPU 间高速通信区域是以 16 字为最小单位的块进行管理。

各机号中可使用的块数、指令中使用的块数如下表所示。

CPU 个数	系统区域 *1	
	1k 点	2k 点
2	46	110
3	22	54
4	14	35

*1 : 关于系统区域设置的有关内容，请参阅 QCPU 用户手册 (多 CPU 系统篇)。

(b) 在由 3 个 CPU 模块组成的多 CPU 系统中，系统区域容量为 1k 字时的多 CPU 间高速通信区域的构成如下所示。



(5) 指令中使用的块数

指令中使用的块数根据进行写入的点数而有所不同。

指令中使用的块数如下表所示。

指令中指定的写入 / 读取点数	D(P)_DDWR 指令	D(P)_DDRD 指令
1 ～ 4	1	1
5 ～ 20	2	
21 ～ 36	3	
37 ～ 52	4	
53 ～ 68	5	
69 ～ 84	6	
85 ～ 100	7	

(6) 可同时执行的多 CPU 间高速通信专用指令

在通用型 QCPU 中，可在下述范围内同时执行多 CPU 间高速通信专用指令。

[各号机中可以使用的块数] ≧ [同时执行的指令使用的块数的合计]

由于执行多 CPU 间高速通信专用指令，多 CPU 间高速通信专用指令使用的块数超过了多 CPU 间高速通信区域的总块数的情况下，在该扫描中不执行本指令（变为无处理）而在下一个扫描中再次执行本指令。但是，执行了本指令时，多 CPU 间高速通信区域的空余块数少于 SD796 ～ SD799（多 CPU 间高速通信专用指令最大块数设置）的设置值的情况下，本指令将变为异常结束状态。

(a) 多 CPU 间高速通信区域的空余块数少于多 CPU 间高速通信专用指令使用的块数或 SD796 ～ SD799 的设置值的情况下，多 CPU 间高速通信专用指令的执行可否如下表所示。

SD设置值与空余块数的大小关系 ＼ 指令使用块数[*1]与空余块数的大小关系	指令使用块数[*1] ≦ 空余块数[*2]	指令使用块数[*1] ＞ 空余块数[*2]
SD设置值[*3] ≦ 空余块数[*2]	执行	不执行(变为无处理)
SD设置值[*3] ＞ 空余块数[*2]	异常结束	

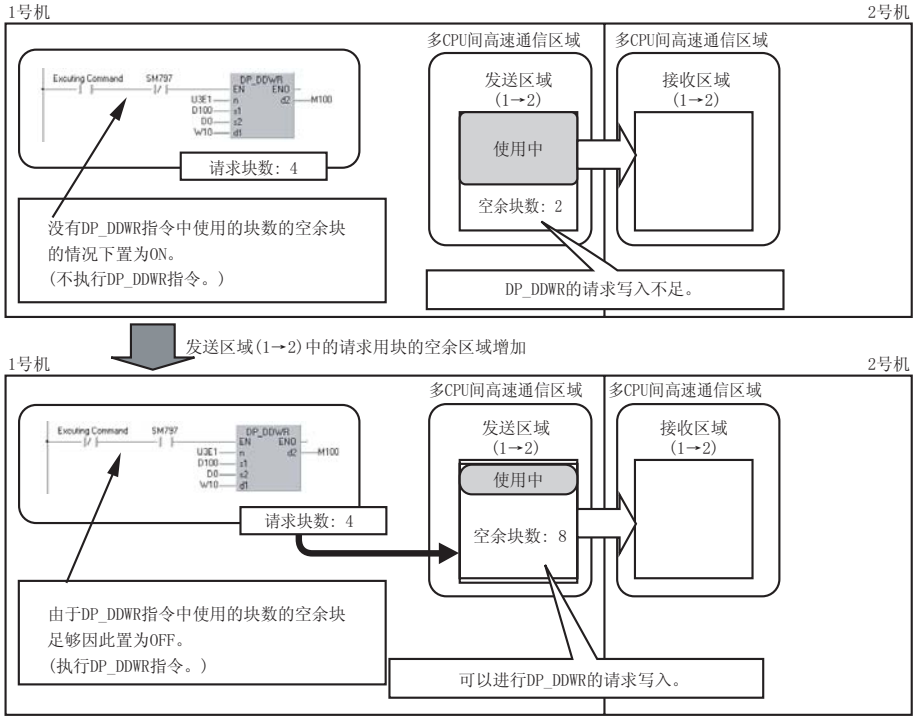
*1: 多CPU间高速通信专用指令使用的块数
*2: 多CPU间高速通信区域的空余块数
*3: SD796～SD799的设置值

(7) 使用多 CPU 间高速通信专用指令时的互锁

(a) 特殊继电器的 SM796 ～ SM799（多 CPU 间高速通信专用指令使用块信息）可作为多 CPU 间高速通信专用指令的互锁用。
同时执行多个多 CPU 间高速通信专用指令的情况下，应将 SM796 ～ SM799 作为多 CPU 间高速通信专用指令的互锁使用。

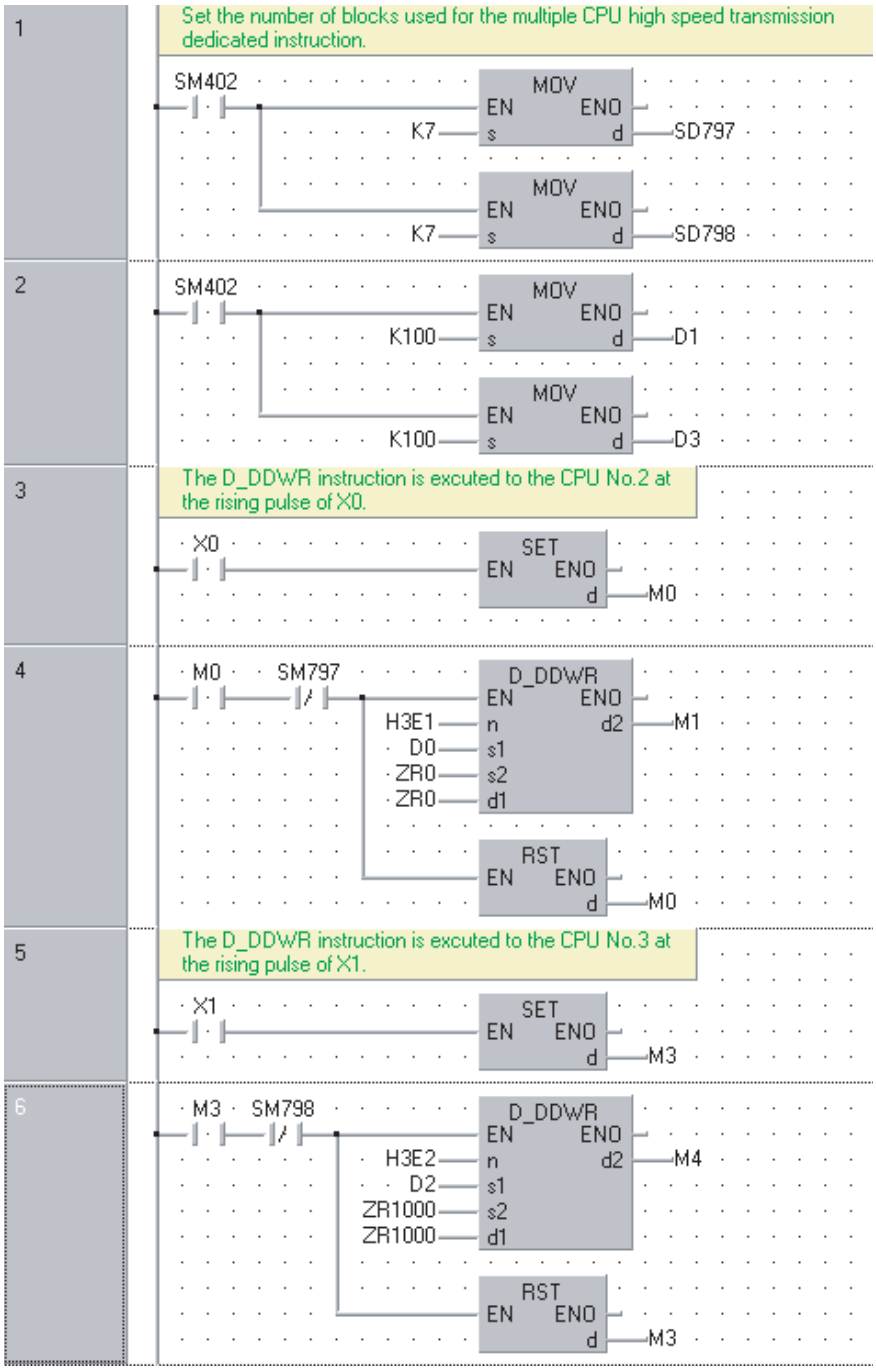
要 点

使用特殊继电器的 SM796 ～ SM799 的情况下，应在特殊寄存器的 SD796 ～ SD799 中设置各号机中使用的指令的最大块数。（例如，对 3 号机执行的多 CPU 间高速通信专用指令的块数的最大值为 5 的情况下，在 SD798 中设置 5。）
多 CPU 间高速通信区域变为 SD796 ～ SD799 中设置的块数以下时，相应的特殊继电器（SM796 ～ SM799）将变为 ON。



(b) 将 SM796 ～ SM799 作为互锁使用的程序示例

在 X0 的上升沿时对 2 号机执行 D_DDWR 指令，在 X1 的上升沿时对 3 号机执行 D_DDWR 指令的程序如下所示。



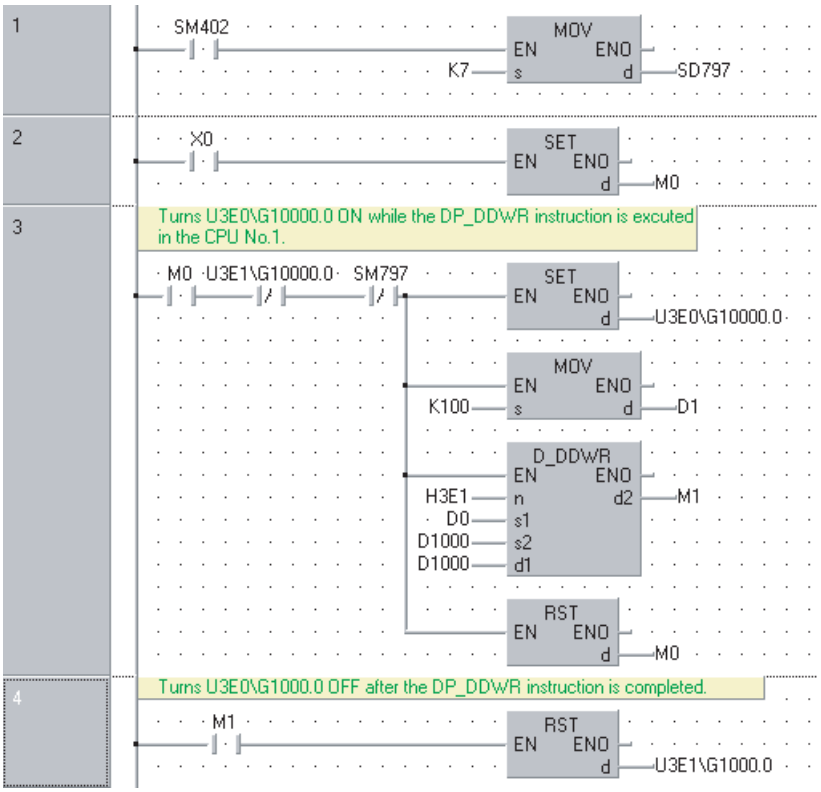
(8) 在多个 CPU 模块中相互执行多 CPU 间高速通信专用指令时的程序示例

在通用型 QCPU 之间，相互执行多 CPU 间高速通信专用指令的情况下，应采取互锁以避免同时执行多 CPU 间高速通信专用指令。

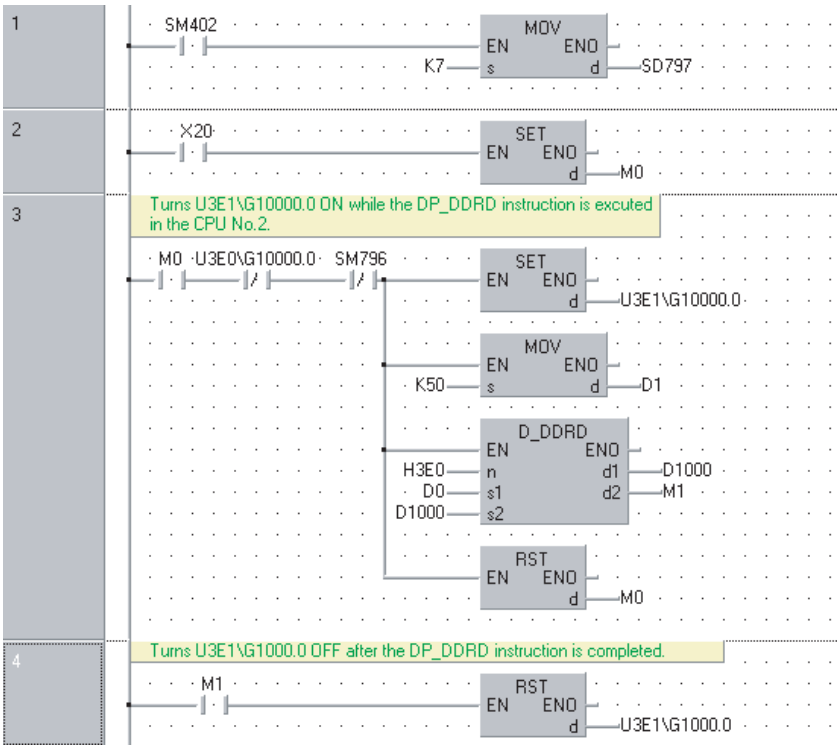
互锁使用多 CPU 间共享软元件 (U3En\G10000 ～)。

1 号机与 2 号机之间相互执行多 CPU 间高速通信专用指令时的程序例如下所示。

在 1 号机中执行多 CPU 间高速通信专用指令时的程序示例



在 2 号机中执行多 CPU 间高速通信专用指令时的程序示例



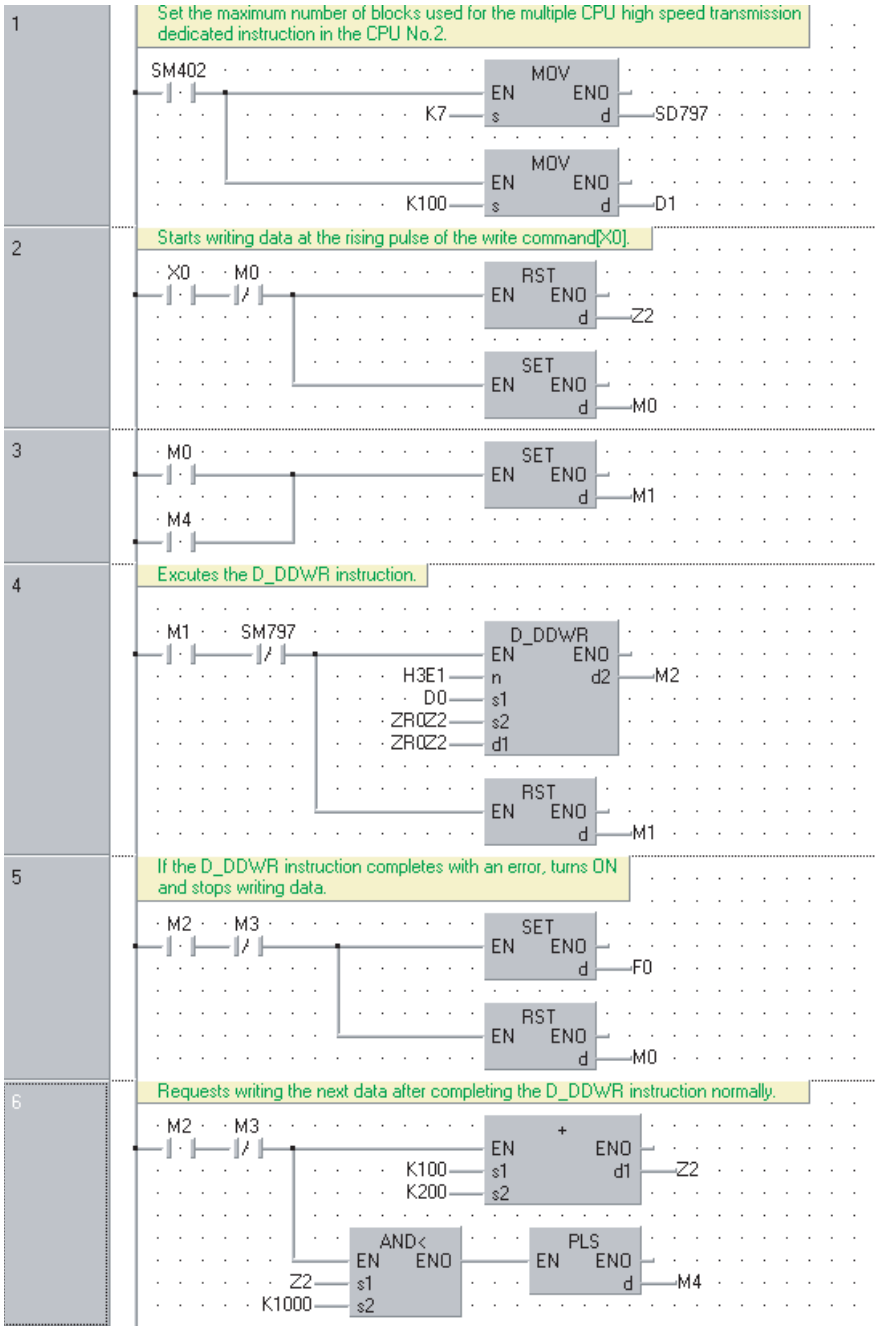
(9) 通过多 CPU 间高速通信专用指令对超过 100 字的数据进行写入 / 读取时的程序示例

通过多 CPU 间高速通专用信指令可处理的写入 / 读取点数最多为 100 字。
对于超过 100 字的数据的写入 / 读取，可通过多次执行多 CPU 间高速通信专用指令来实现。
此外，下图中为使用了多 CPU 间高速通信专用指令的 D(P)_DDWR 指令的程序示例，在使用多 CPU 间高速通信专用指令的 D(P)_DDRD 的情况下，可以使用与下图的程序示例相同构成的程序。

(a) 只启动一个 D(P)_DDWR 指令的程序示例

使用 D_DDWR 指令，将 1 号机的 ZR0 ～ ZR999(1000 点) 写入到 2 号机的 ZR0 ～ ZR999 中的程序示例如下所示。
在下图的程序示例中，通过 D_DDWR 指令的结束软元件 (M2) 的 ON 来启动下一个 D_DDWR 指令，以实现每次只执行一个 D_DDWR 指令的目的。

只启动一个 D(P)_DDWR 指令的程序示例



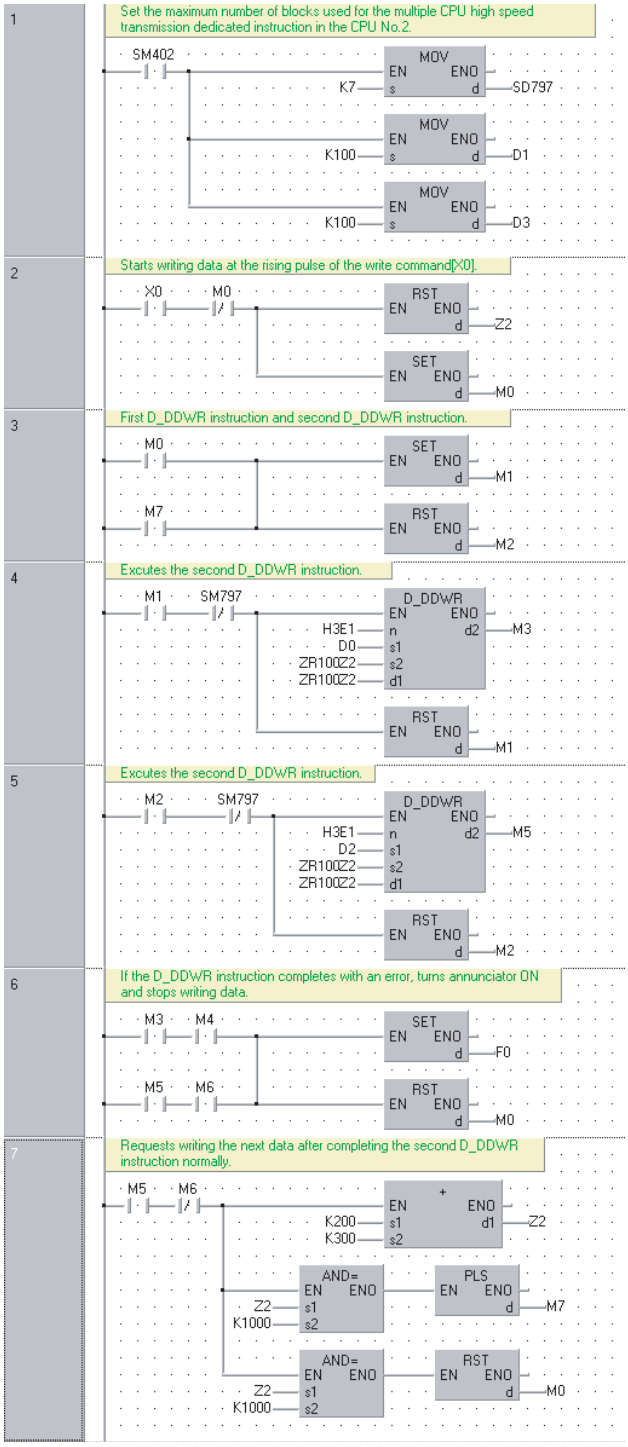
(b) 同时启动二个以上 D(P)_DDWR 指令的程序示例

使用 D_DDWR 指令，将 1 号机的 ZR0 ～ ZR999(1000 点) 写入到 2 号机的 ZR0 ～ ZR999 中的程序示例如下图所示。

如下图程序示例所示，可以同时启动二个以上的多 CPU 间软元件写入 / 读取指令。

同时启动二个以上的多 CPU 间高速通信专用指令的软元件写入 / 读取的情况下，多 CPU 间高速通信区域 (发送区域) 的总块数越多，至多 CPU 间高速通信专用指令的写入 / 读取结束为止的时间就越短。

同时启动多个 D(P)_DDWR 指令的程序示例



10.2 至其它机号的软元件写入

D_DDWR

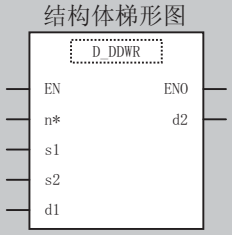
Basic　High performance　Universal　L CPU

Q03UDCPU、Q04UDHCPU、Q06UDHCPU：序列号的前 5 位数为“10012”以后

D(P)_DDWR

P：执行条件　　：

结构体梯形图



ST

ENO:= D_DDWR (EN, n*, s1, s2, d1, d2);

中放入下述指令。

D_DDWR　　DP_DDWR

输入自变量	n:	其它机号 CPU 的起始输入输出编号÷ 1 号机 :3E0H 2 号机 :3E1H 3 号机 :3E2H 4 号机 :3E3H	: ANY16
	s1:	存储控制数据的本站 CPU 的起始软元件	: ANY16 的数组 (0..1)
	s2:	存储写入数据的其它机号 CPU 的起始软元件	: ANY16
	d1:	存储写入数据的本站 CPU 的起始软元件	: ANY*1 : 字符串 *2
输出自变量	d2:	结束软元件	: 位的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它数据
	位	字 *6		位	字			K, H	$	
n*3	–	○	○	–				○	–	–
ⓢ1 *4	–	△ *5	△ *5	–				–	–	–
ⓢ2 *4	–	○	○	–				–	–	–
ⓓ1 *4	–	○	○	–				–	–	–
ⓓ2 *4	△ *7	–	△ *5	–				–	–	–

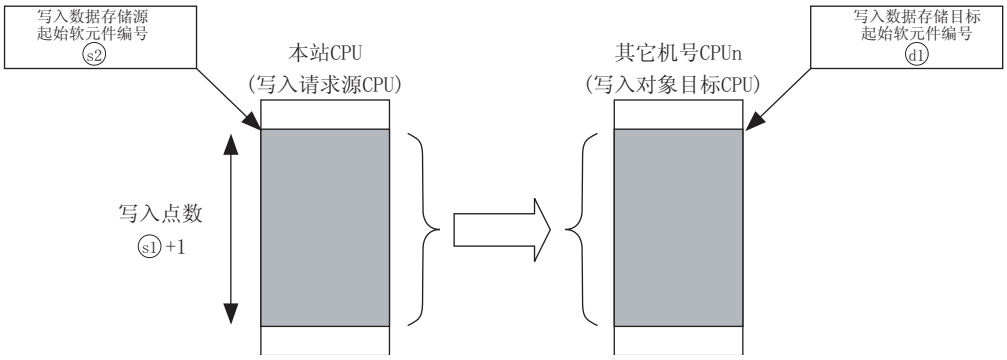
*1 ：指定了文件寄存器 (R、ZR) 的情况下，在本站 CPU 中也可以对范围外的其它机号 CPU 的软元件进行写入。
*2 ：通过将起始软元件以字符串 “ ” 进行指定，在执行本指令的本站 CPU 中也可以对范围外的其它机号 CPU 的软元件进行写入。
*3 ：对于设置数据 n，不能机械能变址修饰。
*4 ：对于设置数据 ⓢ1 ～ ⓓ2 ，可以进行变址修饰。
*5 ：不能使用局部软元件以及各程序中设置的文件寄存器。
*6 ：不能使用 FD。
*7 ：不能使用 FX、FY。

控制数据

软元件	项目	设置数据	设置范围	设置侧
ⓢ1 +0	结束状态	存储指令结束时的执行结果。 0000(H)：无出错（正常结束） 0000(H) 以外：出错代码（异常结束）	—	系统
ⓢ1 +1	写入数据点数	将写入数据点数以字单位进行设置。	1 ～ 100	用户

功 能

(1) 多 CPU 系统配置时，将本站 CPU 中指定的软元件ⓢ2 以后的数据，以ⓢ1 +1 中指定的写入数据点数存储到其它号机 CPU(n) 的指定的软元件ⓓ1以后。



(2) 对于 D(P)_DDWR 指令的正常 / 异常结束，可以通过结束软元件（ⓓ2 +0)、结束时的状态显示软元件（ⓓ2 +1）进行确认。

(a) 结束软元件（ⓓ2 +0）
在指令结束的扫描的 END 处理中置为 ON，在下一个 END 处理中置为 OFF。

(b) 结束时的状态显示软元件（ⓓ2 +1）
根据指令结束时的状态而置为 ON/OFF。

- 正常结束时 :OFF
- 异常结束时 : 在指令结束的扫描的 END 处理中置为 ON，在下一个 END 处理中置为 OFF。（异常结束时在控制数据（(ⓢ1 +0)：结束状态）中存储出错代码。）

(3) 指令中使用的块数取决于写入数据点数。（参阅 10.1 节）

指令中使用的块数

指令中指定的写入点数	D(P)_DDWR 指令
1 ～ 4	1
5 ～ 20	2
21 ～ 36	3
37 ～ 52	4
53 ～ 68	5
69 ～ 84	6
85 ～ 100	7

(4) 多 CPU 间高速通信区域中没有空余块的情况下，即使执行指令也将变为异常结束。通过在特殊寄存器 (SD796 ～ SD799) 中对指令中使用的块数进行设置，将特殊继电器 (SM796 ～ SM799) 作为互锁使用，可以防止变为异常结束状态。(参阅 10.1 节)

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的其它机号 CPU 有错误时。或者不是可以使用多 CPU 间高速通信专用指令的设置时。
(出错代码 :4350)

- 指定了被设置为预约的机号。
- 指定了未安装的机号。
- 其它机号 CPU 起始输入输出编号÷ 16n 超出了 3E0H ～ 3E3H 的范围。
- 在设置为 “不使用多 CPU 间高速通信功能”的状况下，执行了本指令。
- 在 Q02UCPU 中执行了本指令。
- 指定了本站 CPU。
- 指定了不能执行指令的 CPU。

(1) 本指令不能在 CPU 中执行时。 (出错代码 :4351)
- 其它机号 CPU 不支持本指令。

(1) 软元件数有错误时。 (出错代码 :4352)

(1) 指定了不能使用的软元件时。 (出错代码 :4353)

(1) 以不能处理的字符串指定了软元件时。 (出错代码 :4354)

(1) 写入数据点数 (Ⓢ1 +1) 超出了 0 ～ 100 的范围时 (出错代码 :4355)

在以下情况下将变为异常结束，在结束状态存储软元件 (Ⓢ1 +0) 中指定的软元件中将存储出错代码。

(1) 至对象目标 CPU 的指令请求超出了允许值以上。(多 CPU 间高速通信区域中没有空余块。)
(出错代码 :0010H)

(1) ⓓ1中指定的其它机号 CPU 的软元件是在其它机号 CPU 中不能使用的软元件。或者超出了软元件范围。 (出错代码 :1001H)

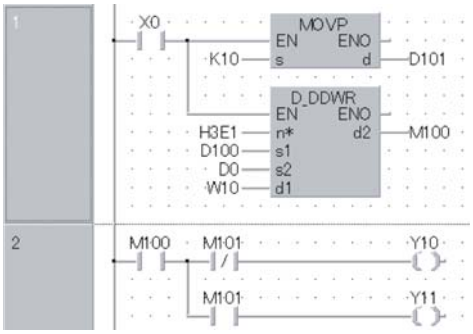
(1) D(P)_DDWR 指令中设置的写入数据点数为 0。 (出错代码 :1080H)

(1) 没有从其它机号 CPU 模块返回指令响应。(多 CPU 间高速通信区域中没有空余块。)
(出错代码 :1003H)

程序示例

以下为 X0 变为 ON 时，将本站 CPU 的 D0 算起的 10 个字的数据存储到 2 号机 CPU 的 W10 以后的程序。

[结构体梯形图]



在控制数据的写入数据点数(ⓢ1+1)设置软元件D101中存储写入数据数“10”。

将本站CPU的D0～D9存储到2号机CPU的W10～W19中。

```
[ST]
IF X0=TRUE THEN
     D101:=10;
     D_DDWR(TRUE,H3E1,D100,D0,W10,M100);
END_IF;
IF M100=TRUE THEN
     IF M101=FALSE THEN
          Y10:=TRUE;
     ELSE
          Y11:=TRUE;
     END_IF;
END_IF;
```

注意事项

(1) n、ⓢ2 以及ⓓ1 中可以进行位软元件的位数指定。但是，在ⓢ2 以及ⓓ1 中进行位软元件的位数指定的情况下合必须满足以下条件。

- 必须是 16 位（4 位数）的位数指定。
- 开始位软元件必须是 16(10H) 的倍数。

(2) 应在写入对象 CPU 已启动的状态下执行本指令。
如果在写入对象 CPU 未启动的状态下执行本指令，将变为无处理。

(3) 在执行本指令后，至结束软元件为 ON 之前的期间，如果对设置数据中指定的软元件范围等进行变更，系统中存储的数据（结束状态、结束软元件）将无法正常存储。

(4) SB、SW、SM、SD 中包含有系统信息区域。
通过多 CPU 间高速通信专用指令的 D(P)_DDWR 指令对上述软元件进行写入的情况下，应注意不要损坏系统信息。

10.3 至其它机号的软元件读取

D_DDRD

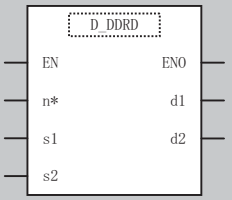
Basic　High performance　Universal　L CPU

Q03UDCPU、Q04UDHCPU、Q06UDHCPU：序列号的前 5 位数为“10012”以后

D(P)_DDRD

P：执行条件　　：

结构体梯形图



ST

```
ENO:= D_DDRD (EN, n*, s1, s2, d1, d2);
```

中放入下述指令。

D_DDRD　　DP_DDRD

输入自变量　n：　其它机号 CPU 的起始输入输出编号÷ 16　：ANY16
1 号机 :3E0H 2 号机 :3E1H 3 号机 :3E2H 4 号机 :3E3H

s1：　存储控制数据的本站 CPU 的起始软元件　：ANY16 的数组 (0..1)

s2：　存储读取数据的其它机号 CPU 的起始软元件　：ANY16

输出自变量　d1：　存储读取数据的本站 CPU 的起始软元件　：ANY*1
：字符串 *2

d2：　结束软元件　：位的数组 (0..1)

设置数据	内部软元件		R, ZR	J□\□		U□\G□	Zn	常数		其它数据
	位	字 *6		位	字			K, H	$	
n*3	–	○	○	–				○	–	–
ⓢ1 *4	–	△ *5	△ *5	–				–	–	–
ⓢ2 *4	–	○	○	–				–	–	–
ⓓ1 *4	–	○	○	–				–	–	–
ⓓ2 *4	△ *7	–	△ *5	–				–	–	--

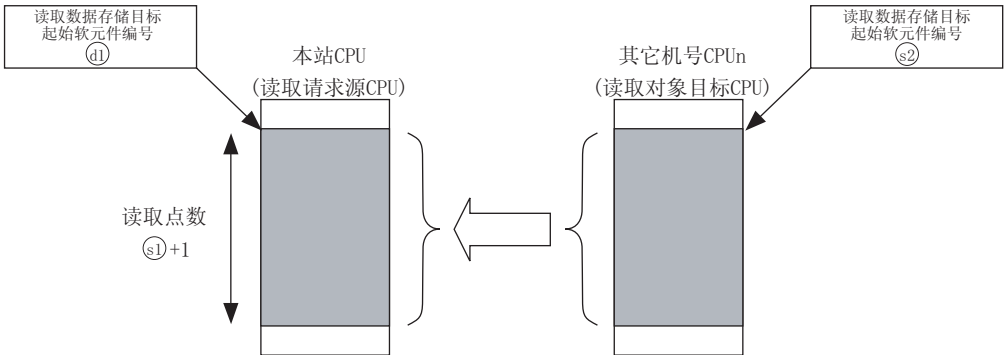
*1 ：指定了文件寄存器 (R、ZR) 的情况下，在本站 CPU 中也可以对范围外的其它机号 CPU 的软元件进行读取。
*2 ：通过将起始软元件以字符串 “ ” 进行指定，在执行本指令的本站 CPU 中也可以对范围外的其它机号 CPU 的软元件进行读取。
*3 ：对于设置数据 n，不能机械能变址修饰。
*4 ：对于设置数据 ⓢ1 ～ ⓓ2 ，可以进行变址修饰。
*5 ：不能使用局部软元件以及各程序中设置的文件寄存器。
*6 ：不能使用 FD。
*7 ：不能使用 FX、FY。

控制数据

软元件	项目	设置数据	设置范围	设置侧
ⓢ1 +0	结束状态	存储指令结束时的执行结果。 0000(H) : 无出错（正常结束） 0000(H) 以外: 出错代码（异常结束）	—	系统
ⓢ1 +1	读取数据点数	将读取数据点数以字单位进行设置。	1 ～ 100	用户

功 能

(1) 多 CPU 系统配置时，将其它机号 CPU(n) 的指定的软元件ⓢ2以后的数据，以ⓢ1 +1 中指定的读取数据点数存储到本站 CPU 中指定的软元件ⓓ1以后。



(2) 对于 D(P)_DDRD 指令的正常 / 异常结束，可以通过结束软元件（ⓓ2 +0）、结束时的状态显示软元件（ⓓ2 +1）进行确认。

(a) 结束软元件（ⓓ2 +0）
在指令结束的扫描的 END 处理中置为 ON，在下一个 END 处理中置为 OFF。

(b) 结束时的状态显示软元件（ⓓ2 +1）
根据指令结束时的状态而置为 ON/OFF。

- 正常结束时 :OFF
- 异常结束时 : 在指令结束的扫描的 END 处理中置为 ON，在下一个 END 处理中置为 OFF。（异常结束时在控制数据（(ⓢ1 +0): 结束状态）中存储出错代码。）

(3) 指令中使用的块数取决于写入数据点数。（参阅 10.1 节）

指令中使用的块数

指令中指定的读取点数	D(P)_DDRD 指令
1 ～ 100	1

(4) 多 CPU 间高速通信区域中没有空余块的情况下，即使执行指令也将变为异常结束。通过在特殊寄存器 (SD796 ～ SD799) 中对指令中使用的块数进行设置，将特殊继电器 (SM796 ～ SM799) 作为互锁使用，可以防止变为异常结束状态。（参阅 10.1 节）

出 错

在以下情况下将发生运算出错，出错标志 (SM0) 将 ON，出错代码将被存储到 SD0 中。

(1) 指定的其它机号 CPU 有错误时。或者不是可以使用多 CPU 间高速通信专用指令的设置时。
(出错代码 :4350)

- 指定了被设置为预约的机号。
- 指定了未安装的机号。
- 其它机号 CPU 起始输入输出编号÷ 16n 超出了 3E0H ～ 3E3H 的范围。
- 设置为 “不使用多 CPU 间高速通信功能”的状况下，执行了本指令。
- 在 Q02UCPU 中执行了本指令。
- 指定了本站 CPU。
- 指定了不能执行指令的 CPU。

(1) 本指令不能在 CPU 中执行时。 (出错代码 :4351)
- 其它机号 CPU 不支持本指令。

(1) 软元件数有错误时。 (出错代码 :4352)

(1) 指定了不能使用的软元件时。 (出错代码 :4353)

(1) 以不能处理的字符串指定了软元件时。 (出错代码 :4354)

(1) 写入数据点数 (①+1) 超出了 0 ～ 100 的范围时 (出错代码 :4355)

在以下情况下将变为异常结束，在结束状态存储软元件 (①+0) 中指定的软元件中将存储出错代码。

(1) 至对象目标 CPU 的指令请求超出了允许值以上。(多 CPU 间高速通信区域中没有空余块。)
(出错代码 :0010H)

(1) ② 中指定的其它机号 CPU 的软元件是在其它机号 CPU 中不能使用的软元件。或者超出了软元件范围。 (出错代码 :1001H)

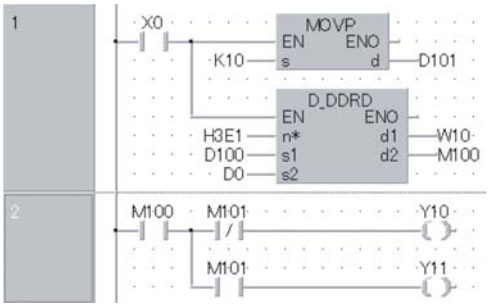
(1) D(P)_DDRD 指令中设置的写入数据点数为 0。 (出错代码 :1081H)

(1) 没有从其它机号 CPU 模块返回指令响应。(多 CPU 间高速通信区域中没有空余块。)
(出错代码 :1003H)

程序示例

以下为 X0 变为 ON 时，将 2 号机 CPU 的 D0 算起的 10 个字的数据存储到本站 CPU 的 W10 以后的程序。

[结构体梯形图]



在控制数据的读取数据点数(①+1)设置软元件D101中存储读取数据数“10”。

将2号机CPU的D0～D9存储到本站CPU的W10～W19中。

```
[ST]
IF X0=TRUE THEN
     D101:=10;
     D_DDRD(TRUE,H3E1,D100,D0,W10,M100);
END_IF;
IF M100=TRUE THEN
     IF M101=FALSE THEN
          Y10:=TRUE;
     ELSE
          Y11:=TRUE;
     END_IF;
END_IF;
```

注意事项

(1) n、㉃以及ⓓ中可以进行位软元件的位数指定。但是，在㉃以及ⓓ中进行位软元件的位数指定的情况下合必须满足以下条件。

- 必须是 16 位（4 位数）的位数指定。
- 开始位软元件必须是 16(10H) 的倍数。

(2) 应在读取对象 CPU 已启动的状态下执行本指令。
如果在读取对象 CPU 未启动的状态下执行本指令，将变为无处理。

(3) 在执行本指令后，至结束软元件为 ON 之前的期间，如果对设置数据中指定的软元件范围等进行变更，系统中存储的数据（结束状态、结束软元件）将无法正常存储。

附 附录

附.1 根据 GX Works2 的版本新增 / 变更的指令

由于 GX Works2 的版本升级而被新增 / 变更的指令如下所示。

附表.1 新增 / 变更指令列表

对应版本	指令名	新增 / 变更内容	参阅
Version1.15R	LDP	・在结构体梯形图中新增了符号表示。	5.1.2 项
	LDF		
	ANDP		
	ANDF		
	ORP		
	ORF		
	LDPI	・新增到结构体梯形图、ST 中。	5.1.3 项
	LDFI		
	ANDPI		
	ANDFI		
	ORPI		
	ORFI		
Version1.24A	UMSG	・新增到结构体梯形图，ST 中。	7.18.20 项

索引

索引

索引

质保

使用之前请确认以下产品质保的详细说明。

1. 免费质保期限和免费质保范围

在免费质保期内使用本产品时如果出现任何属于三菱责任的故障或缺陷（以下称“故障”），则经销商或三菱服务公司将负责免费维修。

注意如果需要在国内现场或海外维修时，则要收取派遣工程师的费用。对于涉及到更换故障模块后的任何再试运转、维护或现场测试，三菱将不负任何责任。

[免费质保期限]

免费质保期限为自购买日或货到目的地日的一年内。

注意产品从三菱生产并出货之后，最长分销时间为 6 个月，生产后最长的免费质保期为 18 个月。维修零部件的免费质保期不得超过修理前的免费质保期。

[免费质保范围]

(1) 范围局限于按照使用手册、用户手册及产品上的警示标签规定的使用状态、使用方法和使用环境正常使用的情况下。

(2) 以下情况下，即使在免费质保期内，也要收取维修费用。

1 因不适当存储或搬运、用户粗心或疏忽而引起的故障。因用户的硬件或软件设计而导致的故障。

2 因用户未经批准对产品进行改造而导致的故障等。

3 对于装有三菱产品的用户设备，如果根据现有的法定安全措施或工业标准要求配备必需的功能或结构后本可以避免的故障。

4 如果正确维护或更换了使用手册中指定的耗材（电池、背光灯、保险丝等）后本可以避免的故障。

5 因火灾或异常电压等外部因素以及因地震、雷电、大风和水灾等不可抗力而导致的故障。

6 根据从三菱出货时的科技标准还无法预知的原因而导致的故障。

7 任何非三菱或用户责任而导致的故障。

2. 产品停产后的有偿维修期限

(1) 三菱在本产品停产后的 7 年内受理该产品的有偿维修。
停产的消息将以三菱技术公告等方式予以通告。

(2) 产品停产后，将不再提供产品（包括维修零件）。

3. 海外服务

在海外，维修由三菱在当地的海外 FA 中心受理。注意各个 FA 中心的维修条件可能会不同。

4. 意外损失和间接损失不在质保责任范围内

无论是否在免费质保期内，对于任何非三菱责任的原因而导致的损失、机会损失、因三菱产品故障而引起的用户利润损失、无论能否预测的特殊损失和间接损失、事故赔偿、除三菱以外产品的损失赔偿、用户更换设备、现场机械设备的再调试、运行测试及其它作业等，三菱将不承担责任。

5. 产品规格的改变

目录、手册或技术文档中的规格如有改变，恕不另行通知。

6. 产品应用

(1) 在使用三菱 MELSEC 通用可编程控制器时，应该符合以下条件：即使在可编程控制器设备出现问题或故障时也不会导致重大事故，并且应在设备外部系统地配备能应付任何问题或故障的备用设备及失效保险功能。

(2) 三菱通用可编程控制器是以一般工业用途等为对象设计和制造的。因此，可编程控制器的应用不包括那些会影响公共利益的应用，如核电厂和其它由独立供电公司经营的电厂以及需要特殊质量保证的应用如铁路公司或用于公用设施目的的应用。

另外，可编程控制器的应用不包括航空、医疗应用、焚化和燃烧设备、载人设备、娱乐及休闲设施、安全装置等与人的生命财产密切相关以及在安全和控制系统方面需要特别高的可靠性时的应用。

然而，对于这些应用，假如用户咨询当地三菱代表机构，提供有特殊要求方案的大纲并提供满足特殊环境的所有细节及用户自主要求，则可以进行一些应用。

MELSEC-Q/L结构体编程手册

公共指令篇

三菱电机自动化(中国)有限公司

地址：上海市黄浦区南京西路288号创兴金融中心17楼
邮编：200003
电话：021-23223030　　传真：021-23223000
网址：www.meas.cn

书号	SH(NA)-080904CHN-A(1004)STC
印号	STC-MELSEC-Q/L(CI)-S-PM(1004)